DEPARTMENT OF DEFENCE
DEFENCE SCIENCE & TECHNOLOGY ORGANISATION
DSTO

# Integrating an Airborne L-band Sea Clutter Model into Research Laboratory Space Time Adaptive Processing (RLSTAP)

Poh Lian Choong

DSTO-TR-1206

# Integrating an Airborne L-band Sea Clutter Model into Research Laboratory Space Time Adaptive Processing (RLSTAP)

*Poh Lian Choong*

**Surveillance Systems Division**
**Electronics and Surveillance Research Laboratory**

**DSTO–TR–1206**

## ABSTRACT

Research Laboratory Space Time Adaptive Processing (RLSTAP) is a compilation of a multiple-channel radar environment, signal and hardware programs integrated within the KHOROS environment. **RLSTAP** was initially developed for the US DARPA Mountain Top Program, and has since been used by several TTCP subgroups for collaborative efforts. Member nations of the SEN TP3 group have been using **RLSTAP** as the baseline radar system simulator for collaborative radar performance assessment. Under SEN TP3 Task 4 *Collaborative Radar Simulation Validation*, we have integrated an L-band sea clutter model with existing RLSTAP programs without extensive redesigning or recoding of the original sea clutter "C" program. This report serves to illustrate how the KHOROS software development tools can be used to modify a stand-alone L-band sea clutter model written in "C" for KHOROS compliance. The rest of the report contains an overview of **RLSTAP** clutter models and detail formulation of the second-order L-band sea clutter model.

**APPROVED FOR PUBLIC RELEASE**

DEPARTMENT OF DEFENCE
DEFENCE SCIENCE & TECHNOLOGY ORGANISATION DSTO

AQ F02-04-0469

*Published by*

*DSTO Electronics and Surveillance Research Laboratory*
*PO Box 1500*
*Salisbury, South Australia, Australia 5108*

*Telephone:* *(08) 8259 5555*
*Facsimile:* *(08) 8259 6567*

*© Commonwealth of Australia 2001*
*AR No. AR-012-013*
*September 2001*

# Integrating an Airborne L-band Sea Clutter Model into Research Laboratory Space Time Adaptive Processing (RLSTAP)

## EXECUTIVE SUMMARY

This work has been performed under the task DST99/045-"RADAR SYSTEMS MODELLING/PREDICTION /VALIDATION" for TTCP SEN-TP3 Task 4 *Collaborative Radar Simulation Validation.*

Research Laboratory Space Time Adaptive Processing (RLSTAP) is a compilation of a multiple-channel radar environment, signal and hardware programs integrated within the KHOROS environment. The KHOROS software development tools provide a complete environment for developing new toolboxes and programs. These programs are graphically represented as glyphs in the KHOROS environment and were written exclusively for modelling monostatic and bistatic radar (**RLSTAP** models use a graphical representation where the radar is functionally represented by hierarchical symbols known as glyphs). A radar model can be constructed graphically by placing and connecting glyphs of the radar hardware, environment and signal processing similar in principle to an operational radar system. Not limited to just constructing simple radar models, the functionality available in **RLSTAP** allows the user to model complex radar systems in defined geographical locations (site-specific clutter). **RLSTAP** is a time based simulation in which the positions of the platform and targets change from pulse to pulse, providing excellent simulation of range (slant range) and Doppler walk. In view of the high fidelity simulation, **RLSTAP** requires significant effort in understanding radar system details and computation resources to develop a workable and realistic radar model.

This report provides the **RLSTAP** user with an introduction to the software development tools currently available within the KHOROS environment to facilitate the integration of non-KHOROS programs into the KHOROS environment. The KHOROS environment has a complete set of software development tools for modifying existing stand-alone program written in either "C" or "C++" for KHOROS compliance. Associated manuals and help documents are automatically generated when KHOROS programs are created. These documents can be edited manually to provide additional descriptions and examples to the user.

We have modified an existing stand-alone modified composite (**MC**) sea clutter program written in "C" for KHOROS compliance and integrated the model with existing **RLSTAP** programs with minimum code modification. This exercise has allowed us to examine the ability to add functionality and to further enhance **RLSTAP** modelling capabilities. The **MC** sea clutter model when incorporated into the KHOROS environment provides a more accurate modelling of the microwave returns from the sea surface for radar operating in the L-band wavelengths.

# Contents

# Tables

# Glossary

$\psi$ grazing angle (to horizontal axis, degrees)

$\theta_i$ incidence angle (to vertical axis, degrees) $= (90° - \psi)$

$\lambda$ RF wavelength of transmitted signal (m)

$\sigma^{\circ}$ mean clutter radar cross-section per unit area of illuminated surface $(dBm^2m^{-2})$

$f$ frequency

**Cantata** A visual programming environment workspace within the KHOROS system.

**Composer** Editing and management of KHOROS software objects.

**Craftsman** Create and manage KHOROS toolbox and software objects.

**CLUI** Command Line User Interface

**DEM** Digital Elevation Model

**glyph** graphical representation of program modules in **RLSTAP**

**GUI** Graphical User Interface

**GUISE** Graphical User Interface Specification Editor

**LULC** Land Use Land Cover

**LUT** Look-Up Table

**MC** Modified Composite Sea Clutter Model

**KHOROS** A software integration and development environment

**RLSTAP** Research Laboratory Space Time Adaptive Processing

**TSC** Technology Service Corporation

**TTCP** The Technical Cooperation Panel

# 1 Introduction

This work was performed under the task DST99/045-"RADAR SYSTEMS MODELLING/PREDICTION /VALIDATION" for TTCP SEN-TP3 Task 4 *Collaborative Radar Simulation Validation.*

Research Laboratory Space Time Adaptive Processing (RLSTAP) is a compilation of a multiple-channel radar environment, signal and hardware programs integrated within the KHOROS environment. These programs are graphically represented as glyphs in the KHOROS environment. A radar model can be constructed graphically by placing and connecting glyphs of the radar hardware, environment and signal processing similar in principle to an operational radar system. Not limited to just constructing simple radar models, the functionality currently available in **RLSTAP** allows the user to model complex radar systems in defined geographical locations (site-specific clutter). **RLSTAP** is a time based simulation in which the positions of the platform and targets change from pulse to pulse, providing excellent simulation of range (slant range) and Doppler walk.

This report discusses an approach used to create an L-band sea clutter model in the KHOROS environment and integration of the model with existing **RLSTAP** programs or *kroutines.* The majority of programs in the KHOROS environment are kroutines. Currently, RLSTAP supports two programs for determining clutter scattering from the sea surface: (1) the "RSS Sigma0 Curves" program generates the backscattering coefficient, $\sigma^\circ$ Look-Up Table (LUT) using the Technology Service Corporation (TSC) RSS model; and (2) the "Sigma0 Curve" program generates the $\sigma^\circ$ LUT using either the ERAM model or the Five Parameter model or a user-specified ASCII Look-Up Table.

The above two clutter scattering programs are representative of the clutter models commonly used in the radar community. They are however, not an accurate representation of the L-band sea clutter characteristic. We have developed a second-order modified composite sea clutter model (**MC**), which is based upon the concept of a two-scale sea surface model to predict L-band radar sea backscattering coefficient, $\sigma^\circ$, for varying sea conditions, grazing angles and radar look directions. We have compared the **MC** sea clutter model and commonly used semi-empirical sea clutter models with data collected by the Naval Research Laboratory four-frequency radar system under varying sea conditions and Jet Propulsion Laboratory AirSAR polarimetric synthetic aperture radar images of the North-West of Australia. The results of the comparison have shown that the **MC** sea clutter model better predicts the backscattering coefficient, $\sigma^\circ$, of the sea surface returns for a wide range of grazing angles, radar look directions and sea conditions.

The first two sections of the report briefly describe the **RLSTAP** sea clutter programs and the formulation of the second order L-band **MC** sea clutter model. Section 4 of the report provides the reader with a tutorial like approach to incorporating new glyphs into the KHOROS environment. Readers not interested in creating new glyphs may skip the entire section and proceed to the modelling results described in Section 5. The approach described in this report is by no means the only method available for integrating new functionality with existing **RLSTAP** programs, but rather a practical method for quickly converting existing stand-alone "C" programs into KHOROS kroutines or glyphs.

# 2 RLSTAP Clutter Scattering Models

For a radar system operating close to the coastal region, or inland areas, its detection and tracking capabilities will be severely affected by compounded microwave return signals from the coastal water, foreshore vegetation, land cover vegetation, land cover usage and man-made structures. The clutter returns from the coastal and inland areas will depend on the terrain characteristics e.g., vegetation type, presence of man-made structures, ground surface roughness and many more. One of the attractions in using RLSTAP for radar modelling is the availability of a site-specific scattering module. This capability not only allows for more accurate modelling of the microwave returns from the land and sea surface, but also accounts for terrain masking or shadowing.

The site-specific clutter modelling module allows the user to specify the area of interest by defining the geo-location, e.g., South-West (reference origin) and the North-East latitude (-90,90) and longitude (-180,180) in degrees. Figure 2.1 shows the flow process of the site-specific clutter modelling. The program locates and extracts: (1) Digital Elevation Model (DEM) data; and (2) Land Use Land Cover (LULC) information for the specific region of interest, and processes them.

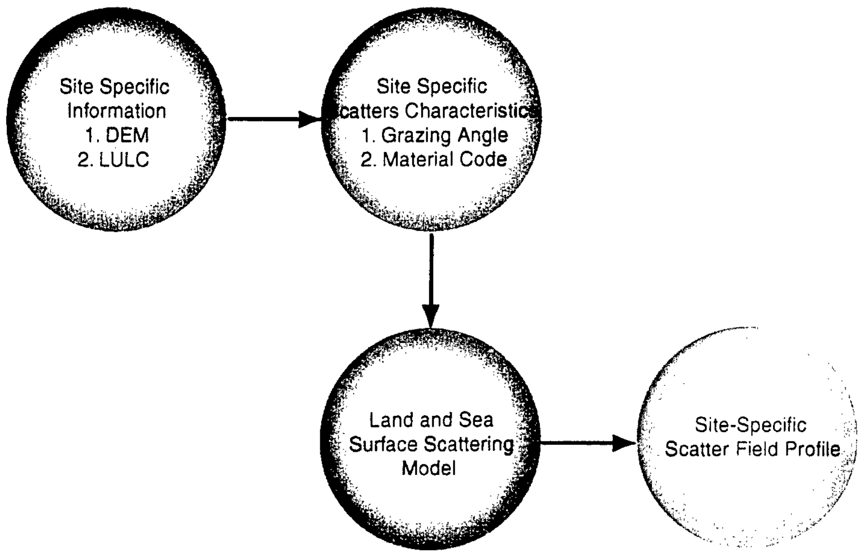


*Figure 2.1: RLSTAP Site Specific Clutter Program Flow*

There are however a few constraints associated with the site-specific module: (1) the resolution of the DEM and LULC data is at present fixed at 3 arc-seconds or approximately 90 m; (2) it only accepts US Geophysical Service (USGS) DEM and LULC data formats; and (3) the sea and land clutter scattering models are limited. The accuracy of modelling the microwave returns from the sea surface and land cover is dependent on the availability of a suitable scattering model in **RLSTAP**.

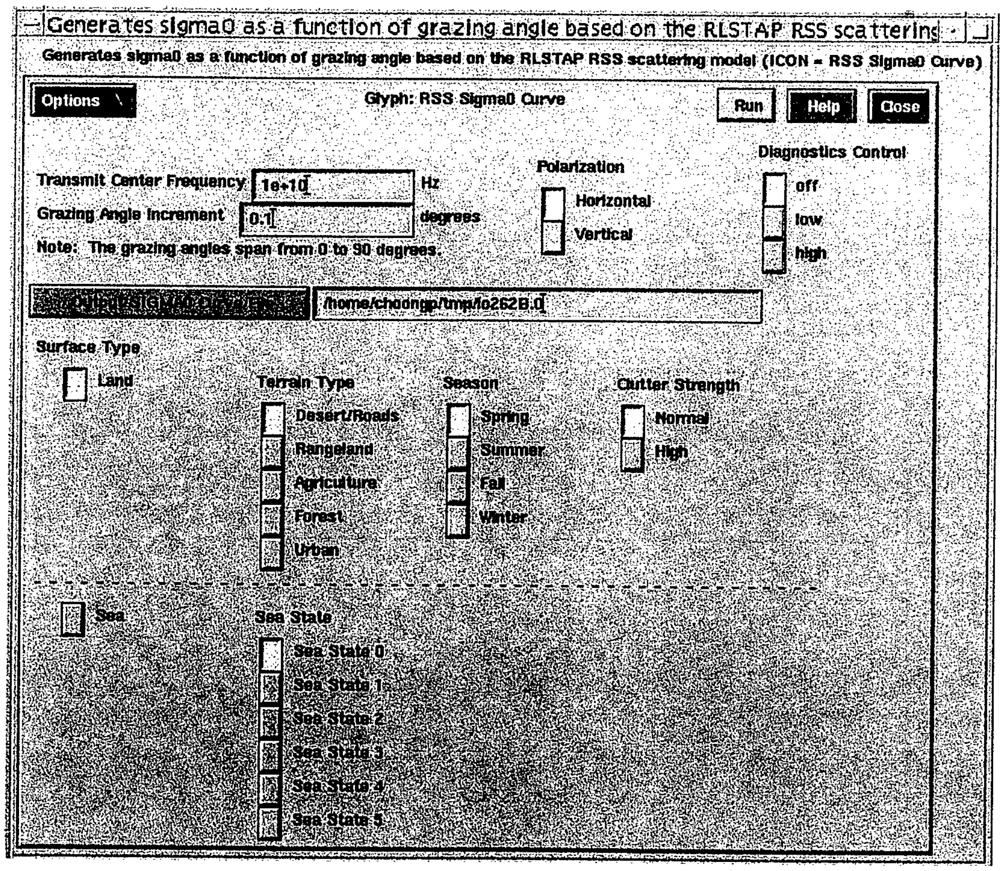
The current **RLSTAP** version 3.13, has two glyphs or programs for defining clutter scattering from the sea surface. Both glyphs generate a LUT of the backscattering coefficient, $\sigma^{\circ}$ versus grazing angle, $\psi$.

## 2.1 RSS Sigma0 Curve

The Graphical User Interface (GUI) for the first clutter scattering program labelled "RSS SIGMA0 Curve" is shown in Figure 2.2. The "RSS SIGMA0 Curve" glyph generates a LUT of the backscattering coefficient, $\sigma^{\circ}$ versus grazing angle, $\psi$, for a specified material code (sea or land cover type) using the TSC RSS model for grazing angles spanning 0 to 90 degrees. In the "RSS Sigma0 Curve" glyph, the user has the option to generate microwave returns from either the land cover or from the sea surface.

For land clutter, five different land cover types are available: (1) Desert/Roads; (2) Rangeland; (3) Agriculture; (4) Forest; and (5) Urban. This scattering model includes the effect of seasonal variation on the strength of the clutter returns.

For sea clutter, six different sea states are available, ranging from sea state 0 to sea state 5.



Generates sigma0 as a function of grazing angle based on the RLSTAP RSS scatterinç

Generates sigma0 as a function of grazing angle based on the RLSTAP RSS scattering model (ICON = RSS Sigma0 Curve)

Glyph: RSS Sigma0 Curve

Transmit Center Frequency 1e+10 Hz

Grazing Angle Increment 0.1 degrees

Note: The grazing angles span from 0 to 90 degrees.

/home/choongp/tmp/lo262B.0

*Figure 2.2: GUI for RSS Sigma0 Curve*

## 2.2 Sigma0 Curve

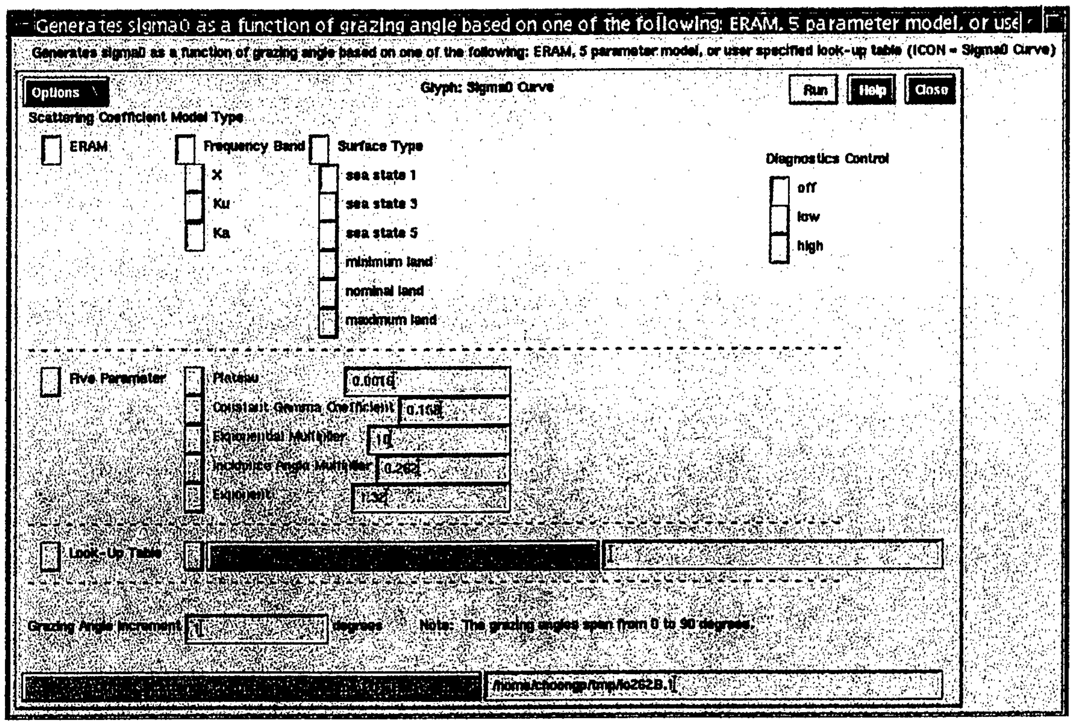
The interface to the second clutter scattering program labelled "SIGMA0 Curve" is shown in Figure 2.3. The "SIGMA0 Curve" glyph generates a LUT of $\sigma^\circ$ as a function of grazing angle, $\psi$ (spanning 0 to 90 degrees), using one of the following clutter models: (1) ERAM, (2) Five parameter model; or (3) user specified ASCII LUT.

In the ERAM model, the user has the option to generate either backscattering from the land cover or from the sea surface for a radar operating in either X, Ka or Ku band. The ERAM model was provided by *Raytheon Systems Company* without any documentation. It is not suitable for modelling radars operating at frequencies less than X-band.

The Five parameter model is independent of frequency, with the backscattering values determined by

$$\sigma^\circ = c_0 + c_1 \sin\psi + c_2 \exp\{-[c_3(90-\psi)]^{c_4}\} \tag{1}$$

where $c_0$ = Plateau, $c_1$= Constant Gamma Multiplier; $c_2$ = Exponential Multiplier; $c_3$ = Incidence Angle Multiplier; and $c_4$ = Exponent.


Generates sigma0 as a function of grazing angle based on one of the following: ERAM, 5 parameter model, or use

Generates sigma0 as a function of grazing angle based on one of the following: ERAM, 5 parameter model, or user specified look-up table (ICON = Sigma0 Curve)

Glyph: Sigma0 Curve

Scattering Coefficient Model Type

ERAM | Frequency Band: X, Ku, Ka | Surface Type: sea state 1, sea state 3, sea state 5, minimum land, nominal land, maximum land

Diagnostics Control: off, low, high

Five Parameter
Plateau: 0.0016
Constant Gamma Coefficient: 0.168
Exponential Multiplier: 10
Incidence Angle Multiplier: 0.262
Exponent: 1.32

Look-Up Table

Grazing Angle Increment: 1 degrees   Note: The grazing angles span from 0 to 90 degrees.

/home/choongp/tmp/lo262B.1

*Figure 2.3: GUI for Sigma0 Curve Model*

The third option is for the user to specify an ASCII file containing a list of grazing angles followed by the corresponding $\sigma^\circ$ values.

# 3 L-band Sea Clutter Model

The principles of microwave scattering by slightly rough surfaces have been applied to backscattering from the sea surface for L-band VV-polarisation under standard atmospheric conditions [2]. Based on the two-scale sea surface model proposed by Guinard and Daley [2] and the surface scattering model proposed by Chen, Fung and Weissman [5], the second-order modified composite (*MC*) sea clutter model takes into account polarisation, wind speed, azimuth and grazing angles variations. In this model, the backscattering coefficient, $\sigma^{\circ}$, can be represented as the sum of separate contributions from the specular return and resonant Bragg-scattering.

Three commonly used sea clutter models: (1) Georgia Institute of Technology (GIT): (2) the Hybrid (HYB); and (3) the Technology Service Corporation (TSC) sea clutter models were also considered for comparison [6]. These sea clutter models are compared with four sets of measured L-band VV-polarised data.

Of the four data sets used, three sets of calibrated airborne L-band data were collected by the Naval Research Laboratory (NRL) in the late 60s and early 70s from several sea locations and in varying sea conditions. The fourth set of data was obtained from measured L-band VV-polarised polarimetric synthetic aperture radar (POLSAR) images of the Darwin region. These images were acquired during the PACRIM mission by the Jet Propulsion Laboratory on the 23rd November 1996.

## 3.1 Data Source for L Band

Four sets of measured L-band VV-polarised data have been used to test the accuracy of the **GIT**, **HYB**, **TSC** and the proposed **MC** sea clutter models. Brief descriptions of the NRL 4FR data sets, and the DARWIN POLSAR data are summarised below.

### 3.1.1 Naval Research Laboratory Airborne Trials [1, 2, 3, 4]

The Naval Research Laboratory (NRL) measurements from the 60s and early 70s used for validating the model were obtained using a four-frequency (4FR) airborne radar system. The carrier frequencies were: (1) UHF (428 MHz); (2) L-band (1.228 GHz); (3) C-band (4.455 GHz), and (4) X-band (8.910 GHz)[7, 3, 4]. For all analyses, the median sea backscattered values were converted to mean values assuming a Rayleigh amplitude distribution [7].

1. NRL-4FR Data [1]

   Sea backscattered measurements were recorded using the NRL 4FR radar system, off New Jersey in 1964, off Puerto Rico in 1965 and in the North Atlantic in 1969. Surface truth information for the Puerto Rico deployment was measured by a team of researchers from the Applied Physics Laboratory, Johns Hopkins University, in an instrumented vessel [2]. For the North Atlantic deployment, the surface truth information was provided by oceanographic surface vessels operated by the French and British governments [2]. We will hereafter refer to data measured as the NRL-4FR data.

2. Joint Ocean Surface Study, 1970 (JOSS I) [4]

   The NRL 4FR system was used to measure sea backscattered signals off the vicinity of Argus Island, Bermuda. Wind and wave height information were recorded by observers stationed on Argus Island.

3. Joint Ocean Surface Study, 1971 (JOSS II) [8]

   The NRL 4FR system was used to measure sea backscattered signals off the East Coast of Northern America. Wind velocity, direction, and average wave height information were obtained hourly from an instrumented buoy and an ocean station vessel. [8].

### 3.1.2 AirSAR Polarimetric Synthetic Aperture (POLSAR) Data

POLSAR is a NASA-sponsored polarimetric synthetic aperture radar system developed at the Jet Propulsion Laboratory in California. The POLSAR system provides high quality polarimetric SAR data in three frequencies; P (0.44 GHz), L (1.25 GHz) and C (5.39 GHz) bands in four transmit/receive polarisations (VV, HH, VH, HV).

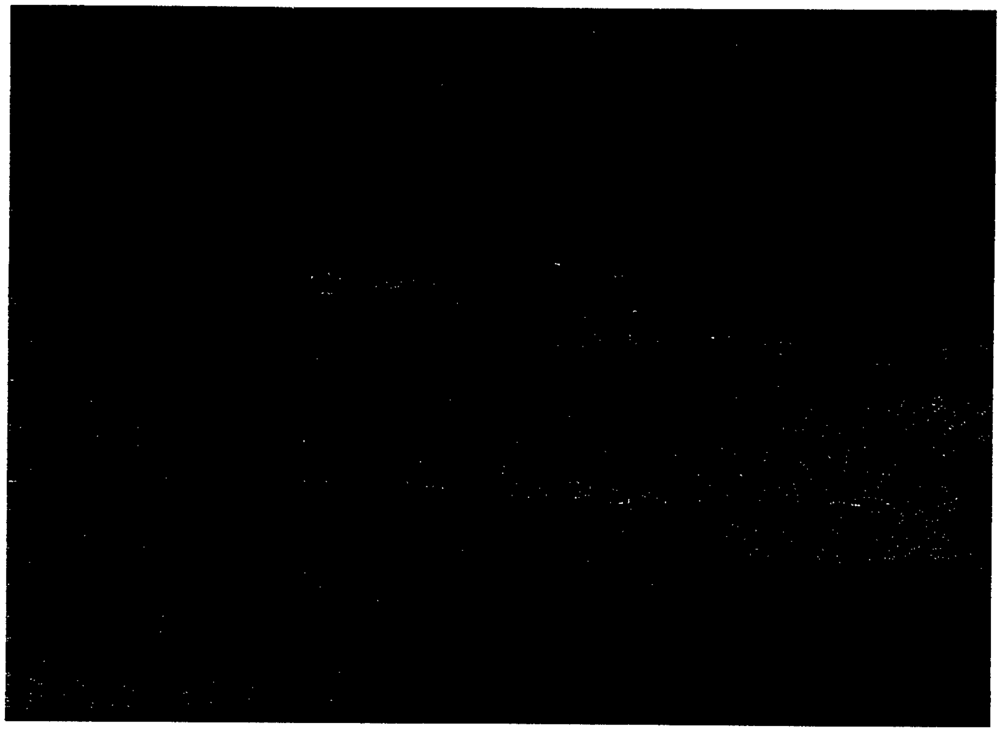
The study area of interest was in the vicinity of Darwin (130° 45 ' E, 12 ° 30' S) in the Northern Territory of Australia. Two flight passes were selected and the two L-band polarimetric SAR images of vertical (VV) polarisation collected with a bandwidth of 40 MHz are shown in Figures 3.1(a) and (b). Specifications of the POLSAR images are tabulated in Table 3.1. The number of looks processed in azimuth is 18, and 1 in range. Recorded measurements of the sea conditions are tabulated in Table 3.2.

*Table 3.1: POLSAR Specifications*

| Description | CM5512 | CM5148 |
|---|---|---|
| Near Slant Range (m) | 9013.26 | 9013.26 |
| Far Slant Range (m) | 17537.36 | 19244.51 |
| Near Look Angle ($\theta_i$) | 22.5° | 22.6° |
| Far Look Angle ($\theta_i$) | 61.7° | 61.7° |
| Range Resolution (m) | 3.3310 | 3.3310 |
| Azimuth Resolution (m) | 9.2592 | 9.2592 |

*Table 3.2: Hydro-meteor Parameters*

| Description | Darwin Harbour | Fannie Bay |
|---|---|---|
| Air Temperature (Dry) | 35.0°C | 32.5°C |
| Air Temperature (Wet) | 29.0°C | 27.0°C |
| Relative Humidity | 68% | 64% |
| Wind Speed | 14 km/h | 4 kt |
| Wind Direction | 170° | SE |
| Wave Height | 1 cm | Calm |


(a) CM5512

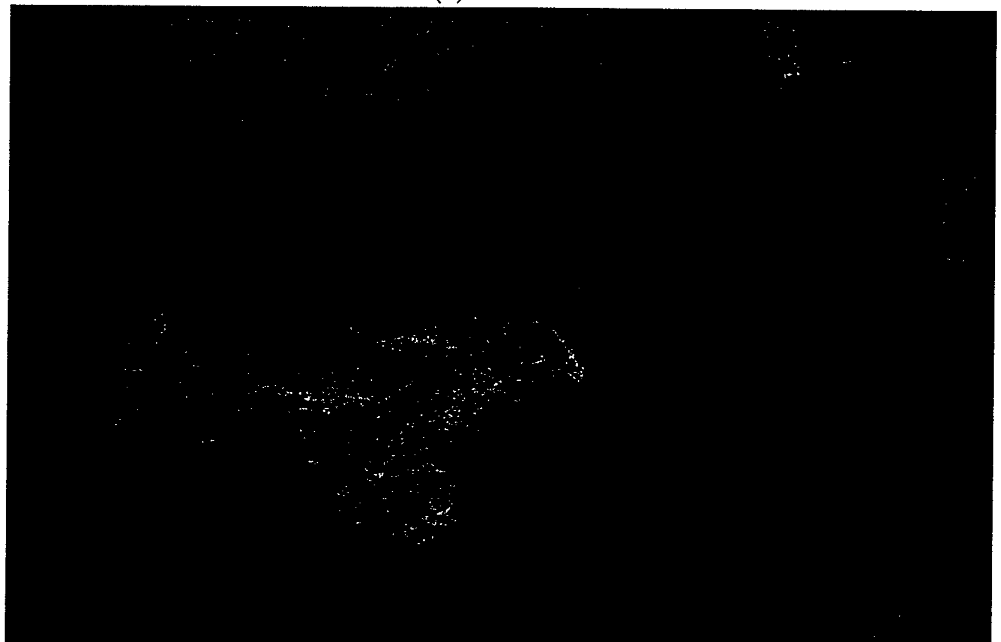

(b) CM5148

*Figure 3.1: POLSAR Darwin L-band VV data*

## 3.2 Modified Composite (MC) Sea Clutter Model

The **MC** sea clutter model is based upon the concept of slightly rough sea surface scattering in which the dominant mechanism is Bragg resonant scattering, whose amplitude is related to the second-order directional sea spectrum proposed by Chen, Fung and Weissman [5]. The model also accounts for specular return near-normal incidence. The backscattering coefficient, $\sigma^\circ$, from the sea surface can be represented as the sum of separate contributions from specular return and Bragg-scattering.

$$\sigma^\circ = \underbrace{\sigma_S^\circ}_{\text{Specular Returns}} + \underbrace{\sigma_B^\circ}_{\text{Bragg Scattering}} \tag{2}$$

The **MC** sea clutter model is constructed based on these two forms of scattering mechanism. The primary scattering mechanism in the low-grazing angle and plateau regions is assumed to be Bragg scattering whereby the backscattering coefficient, $\sigma_B^\circ$, is in proportion to the spectral density of the short gravity-capillary waves for a fully developed sea [9]. At near normal incidence, the effect of specular return, $\sigma_S^\circ$, due to facets normal to the radar signal is included.

### 3.2.1 Specular Backscattering for Fully Developed Sea

Near-normal incidence backscatter from the sea surface is better described by the specular point or tilted facet model. For an isotropic rough surface of Gaussian statistics, the backscatter return at near normal incidence is given by [10]

$$\sigma_S^\circ = \frac{|R(0)|^2}{s^2} \sec^4\theta_i \exp(-\tan^2\theta_i/s^2) \tag{3}$$

where $R(0)$ is the Fresnel reflection coefficient of the sea surface at normal incidence, $s^2$ is the mean-square sea slope and $\theta_i$ is the angle of incidence.

(a) Mean-Square Sea Slope, $s^2$

$$s^2 = \begin{cases} 5.12 \times 10^{-3}U, & U < 5\text{m/s} \\ 1.56 \times 10^{-3}U, & U \geq 5\text{m/s} \end{cases} \tag{4}$$

where $U$ (m/s) is the wind speed at an altitude 12.5 m above the sea surface.

(b) Reflection Coefficient, $|R(0)|$

$$|R(0)| = \left| 0.52 \frac{\epsilon_r - 1}{(\sqrt{\epsilon_r} + 1)^2} \right| \tag{5}$$

where $\epsilon_r$ is the relative dielectric constant of sea water.

### 3.2.2 Bragg Scattering

1. Second Order Bragg Scattering Model, $\sigma^{\circ}_{B^2(pp)}$

$$\text{Horizontal Polarisation}, \sigma^{\circ}_{B^2(HH)} = k_{\circ}^2 \sin^4(90-\theta_i)\,\alpha_{HH} \underbrace{W(K,\phi)}_{\text{Directional Sea Spectrum}} \qquad (6)$$

$$\text{Vertical Polarisation}, \sigma^{\circ}_{B^2(VV)} = k_{\circ}^2 \sin^4(90-\theta_i)\,\alpha_{VV} \underbrace{W(K,\phi)}_{\text{Directional Sea Spectrum}}$$

where $k_{\circ} = \frac{2\pi}{\lambda}$ is the radar wavenumber, $\theta_i$ is the angle of incidence, $K$ is the sea wavenumber, and $\phi$ is the wind direction with respect to boresight.

2. The first-order scattering coefficient, $\alpha_{pp}$ is [2]

$$\text{Horizontal Polarisation}, \alpha_{HH} = \frac{(\epsilon_r - 1)}{[\cos\theta_i + \sqrt{(\epsilon_r - \sin^2\theta_i)}]^2} \qquad (7)$$

$$\text{Vertical Polarisation}, \alpha_{VV} = \frac{(\epsilon_r - 1)[\epsilon_r(1+\sin^2\theta_i) - \sin^2\theta_i]}{[\epsilon_r\cos\theta_i + \sqrt{(\epsilon_r - \sin^2\theta_i)}]^2}$$

where $\epsilon_r$ is the relative dielectric constant of the sea surface with value $72 - j59$ (L-band, 1.275 GHz) [9].

3. Two-dimensional (2-D) Sea Ripple Spectrum

Adopting the 2-D sea ripple spectrum developed by Pierson and modified by Chen and Fung [5], the 2-D directional (taking into account the wind direction) sea spectrum is

$$W(K,\phi) = \frac{1}{2\pi}(1 + r\cos 2\phi)W(K) \qquad (8)$$

(a) The constant, $r$, is given by

$$r = 2\frac{(1-\nu)}{(1+\nu)}$$

where $\nu$ is the ratio of the slope variances in the *crosswind* and *upwind* directions [5],

$$\nu = \frac{s_c^2}{s_u^2} = \begin{cases} \frac{0.003+1.92\times10^{-3}U}{3.16\times10^{-3}U}, & U < 5\text{m/s} \\ \frac{0.003+0.84\times10^{-3}U}{0.005+0.78\times10^{-3}U}, & U \geq 5\text{m/s} \end{cases}$$

(b) Wind Direction, $\phi$

$$\text{Wind direction}, \phi = \begin{cases} 0^\circ, & \text{upwind} \\ 90^\circ\text{or} - 90^\circ, & \text{crosswind} \\ 180^\circ, & \text{downwind} \end{cases}$$

(c) 1-D sea ripple spectra for a single wave train, $W(K)$

$$W(K) = A(2\pi)^{p-1}\frac{\left(1 + 3\frac{K^2}{K_m^2}\right) g^{\frac{(1-p)}{2}}}{K^{0.922}\left[K\left(1 + \frac{K^2}{K_M^2}\right)\right]^{\frac{(1+p)}{2}}} \qquad (9)$$

where $A = 1.75$ if $U < 5$ m/s or $A = 0.875$ if $U \geq 5$ m/s, $K_m^2 = g\rho/\tau = (3.63)^2\text{cm}^{-2}$, $g$ = acceleration of gravity ($9.81\,\text{m/s}^2$), $\rho$ = sea water density, and $\tau$ = sea surface tension. Note that according to Valenzuela [11], K should have a $-3.922$ power law rather than the usual $-4$ power law for L-band resonant Bragg waves.
The relationship between the sea surface wind speed at an altitude $z$ and friction velocity at neutral stability of the atmosphere, $U^*$ is [5]

$$U = \frac{U^*}{0.4} \ln\left(\frac{z}{Z_0}\right)$$

where

$$Z_0(\text{cm}) = \left(\frac{0.684}{U^*}\right) + 4.28 \times 10^{-5} U^{*2} - 0.0443$$

and $p = 5 - \log_{10} U^*$.

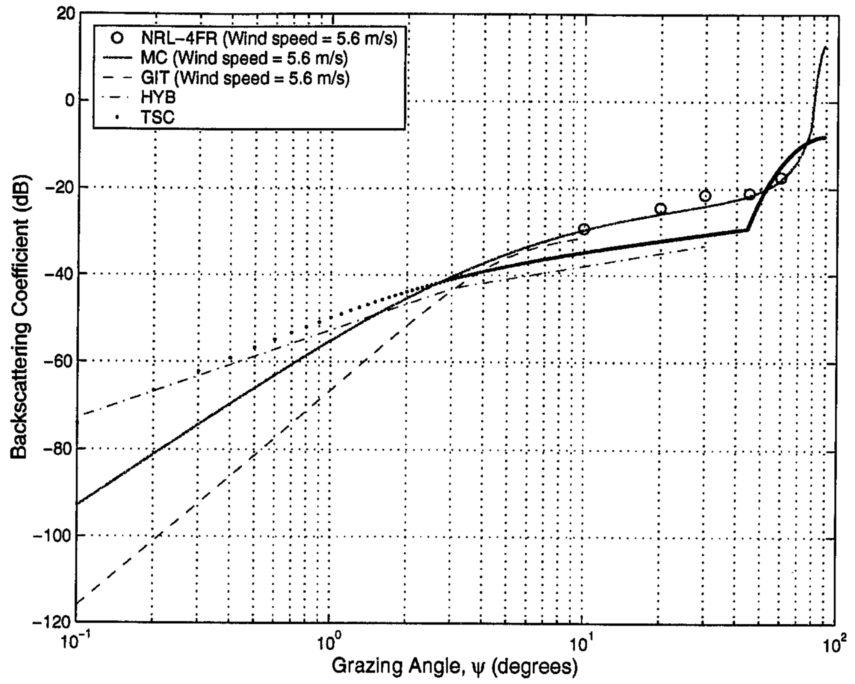
## 3.3 Comparisons with Radar Measurements

Plots of the results in Figures 3.2 (a) and (b) show good agreement between the measured NRL-4FR data and the **MC** predicted $\sigma^\circ$ values for the two wind speeds. Both the **HYB** and **TSC** models underestimate the backscattered values, $\sigma^\circ$.

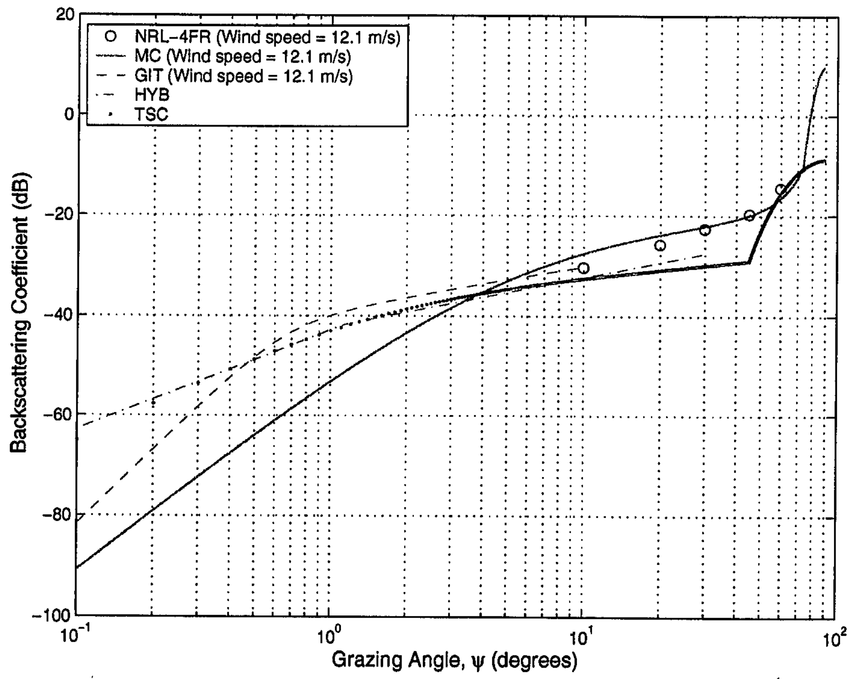
We further tested the validity of the **MC** sea clutter model using the JOSS I, JOSS II, and the POLSAR DARWIN measured data. Figure 3.3(a) again shows the poor agreement of the **TSC** and **HYB** models with the JOSS I measured data. Where these two models fail, the **MC** sea clutter model shows good agreement with the measured data. Figure 3.3(b) shows that the **MC** sea clutter model accurately models the JOSS II measured data. Note that the **GIT** model has good agreement with the NRL-4FR and JOSS I data for limited grazing angles, $\psi \in [5^\circ, 10^\circ]$. However, further validation is required due to the lack of low grazing angle data in the available data sets.

For the POLSAR data acquired over the DARWIN region under low wind speed condition, the **MC** sea clutter model accurately predicted the backscattering coefficient, $\sigma^\circ$, for $20^\circ < \psi < 65^\circ$. Figure 3.3(c) shows the measured data, and the predicted $\sigma^\circ$ values by the **MC** and semi-empirical sea clutter models. The **TSC** and **HYB** models underestimate the backscatter values when compared to the measured data.

We have tested four sea clutter models against measured data. These results suggested that among these four (**GIT**, **HYB**, **TSC**, **MC**) sea clutter models, the **MC** sea clutter model is the preferred generic sea clutter model for predicting the L-band, VV-polarised sea clutter for a wide range of grazing angles.

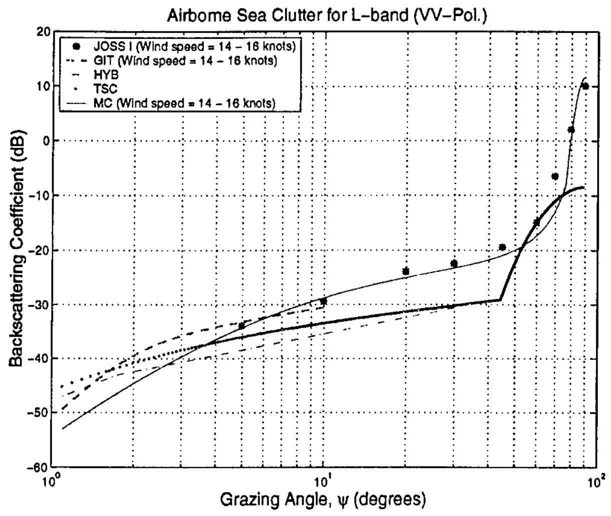


(a) Wind speed, U = 5.6 m/s

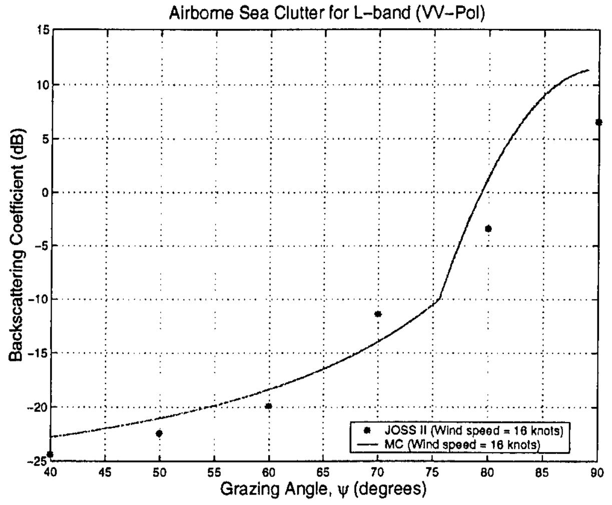


(b) Wind speed, U = 12.1 m/s

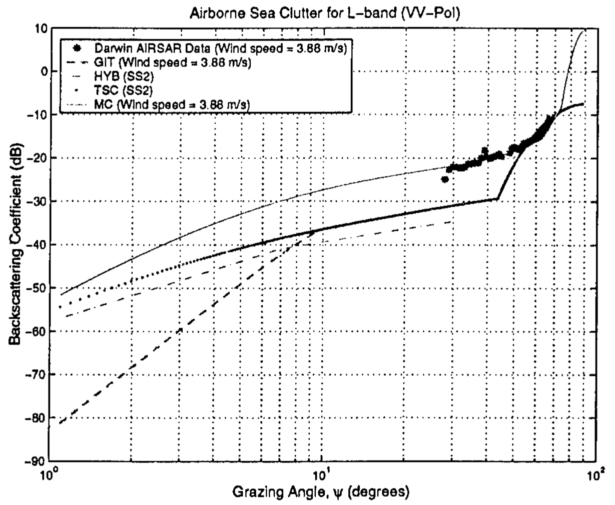
*Figure 3.2: Dependence of $\sigma^{\circ}$ on Wind Speed, U (L-band, VV-Polarisation and Upwind).*



(a) JOSS I



(b) JOSS II



(c) DARWIN POLSAR Data

*Figure 3.3: Dependence of $\sigma^\circ$ on Wind Speed, U (L-band, VV-Pol. and Upwind).*

# 4 L-band MC Sea Clutter Glyph Creation

This section illustrates an application of the KHOROS software development tools to create a new *sea clutter* toolbox to contain the **MC** sea clutter model and a method to convert an existing **MC** sea clutter program, labelled "SeaClutter.c" into a KHOROS program or *kroutine*. In the KHOROS system, a *toolbox* is a collection of programs and libraries grouped together with a common functionality. The majority of these programs are in the standard KHOROS *kroutines* format.

**Step 1. Toolbox Creation**

The first step in inserting a "stand-alone C" code into the KHOROS environment is to create either a new toolbox or use an existing toolbox. This is done using **Craftsman** [12, 13]. The **Craftsman** application is used to create and manage KHOROS toolbox and software objects. Start up **Craftsman** from the command line. When **Craftsman** is displayed (refer to Figure 4.1), you will notice two lists: (1) the list on the left corresponds to the toolboxes currently available; and (2) the right list corresponds to a selected toolbox software objects.

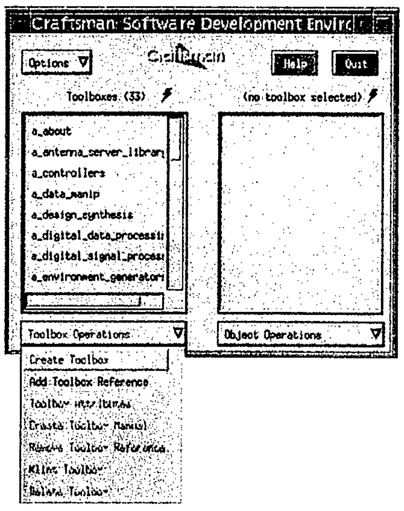
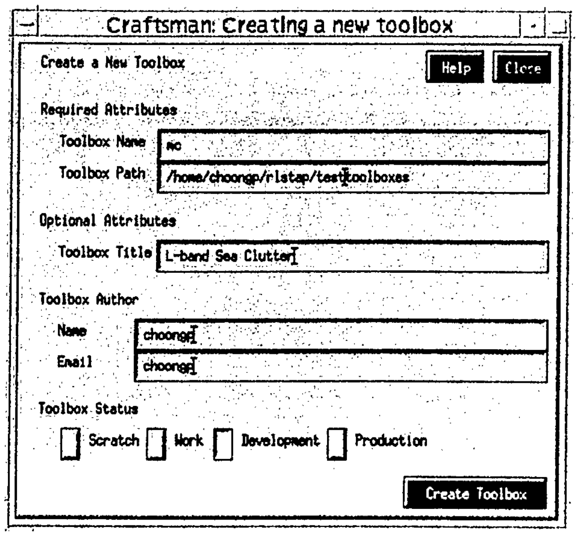


*Figure 4.1: The* **Craftsman** *software development environment*

(a) To create a new *sea clutter* toolbox, select **Create Toolbox** from the **Toolbox Operations** pulldown list.

(b) This action brings up a sub-menu shown in Figure 4.2 and allows you to enter all the software object information.

1. *Toolbox Name*
   Type in the toolbox name as you would like it to appear in **Craftsman**. In this example, the sea clutter toolbox will appear on the **Craftsman** toolboxes list as **MC**.


Craftsman: Creating a new toolbox

Create a New Toolbox

Required Attributes

Toolbox Name: mc

Toolbox Path: /home/choongp/rlstap/testtoolboxes

Optional Attributes

Toolbox Title: L-band Sea Clutter

Toolbox Author

Name: choongp

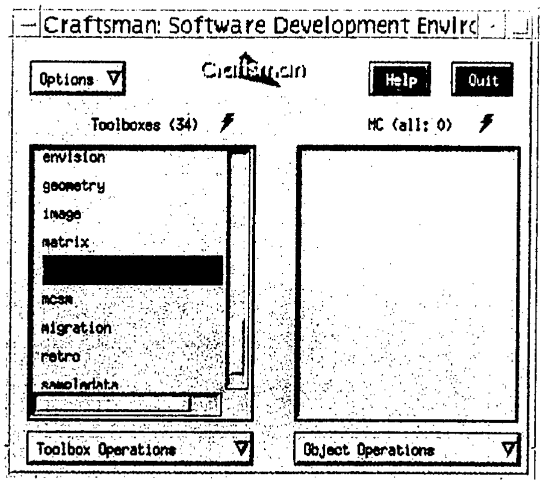
Email: choongp

Toolbox Status: Scratch, Work, Development, Production

*Figure 4.2: The* **Craftsman** *Create Toolbox Sub-menu*

2. *Toolbox Path*
   Directory preference for toolbox location.
3. *Toolbox Title*
   Short description of the toolbox title.
4. *Point of Contact Information*
   The point of contact information contains the author's name and email address, primarily used for bug control reporting for public release KHOROS programs.

(c) Click on the **Create Toolbox** button, and the new sea clutter toolbox labelled **MC** is now created. It will appear in the toolbox list as shown in Figure 4.3.


Craftsman: Software Development Enviro

Toolboxes (34)

envision
geometry
image
matrix
mcsm
migration
retro
sampledata

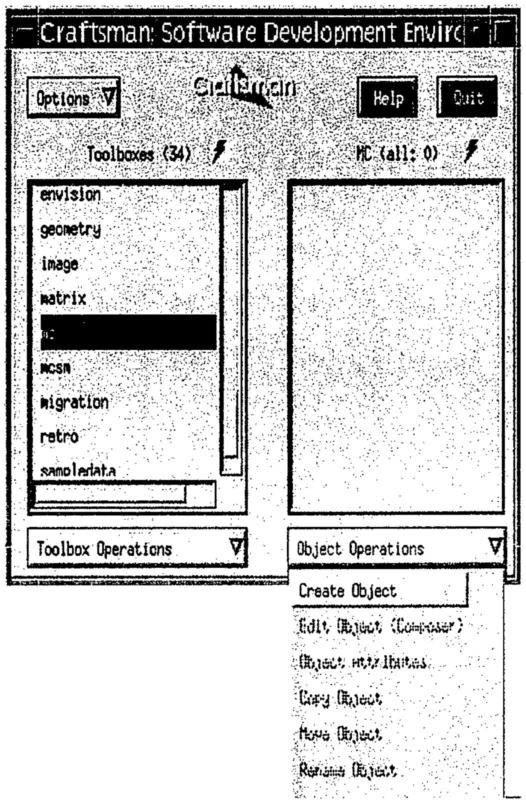
MC (all: 0)

*Figure 4.3: The Newly created Sea Clutter Toolbox labelled* **MC**

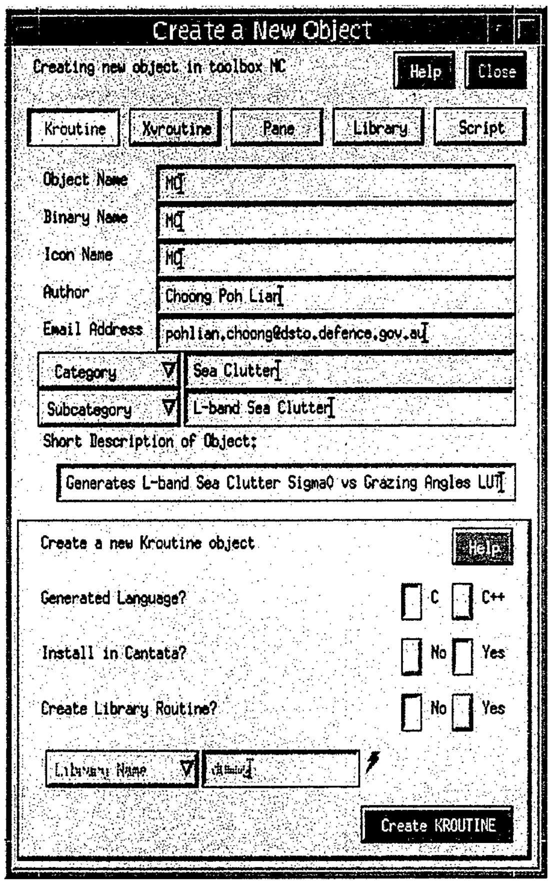
**Step 2. Progam Object Creation**

The next step involves creating the sea clutter KHOROS program object. The program object comprises the **MC** sea clutter program along with a KHOROS imposed directory structure and KHOROS generated code.

(a) Select **Create Object** from the **Object Operations** pulldown list. Selecting this will bring up the object creation submenu as shown in Figure 4.4(b).



(a)



Creating new object in toolbox MC

Object Name: MC

Binary Name: MC

Icon Name: MC

Author: Choong Poh Lian

Email Address: pohlian.choong@dsto.defence.gov.au

Category: Sea Clutter

Subcategory: L-band Sea Clutter

Short Description of Object: Generates L-band Sea Clutter Sigma0 vs Grazing Angles LUT

Create a new Kroutine object

Generated Language? C C++

Install in Cantata? No Yes

Create Library Routine? No Yes

(b)

*Figure 4.4: (a)* **Craftsman** *; and (b) the Object Creation Submenu*

(b) Fill in all the program objects attributes.

1. *Object Name*
   Title of the object and the source code of the **MC** sea clutter program will be placed in a directory of this name.
2. *Binary Name*
   Name of the executable and library archive (if not specified, default to Object Name).

3. *Icon Name*
   Title of glyph which appears on the **Cantata** pulldown list as a subcategory (if not specified, default to Object Name).
4. *Category and Subcategory*
   Use to classify the object to a broad domain application. In this example, the **MC** sea clutter model is classified under a broader category of "Sea Clutter". Within the *Sea Clutter* category, the **MC** sea clutter model falls into the *subcategory* of "L-band Sea Clutter".
5. *Generated Language*
   Choices are "C" or "C++". The existing sea clutter program was written in "C", therefore the "C" option was selected.
6. *Install in Cantata*
   This toggle is used to specify whether you want the object to be visible to **Cantata** .
7. *Create Library Routine*
   This toggle is used to specify whether you want a library object to be associated with your object.

Upon completion, click on the **Create KROUTINE** to create the **MC** sea clutter program object (directory structure, makefiles, source code and others associated with the program object). At this stage, the main KHOROS source file generated for the **MC** sea clutter kroutine is labelled "MC.c" and the object code, "MC.o". When the new **MC** program object has been created, it will appear highlighted in the **Craftsman** object list as shown in Figure 4.5.

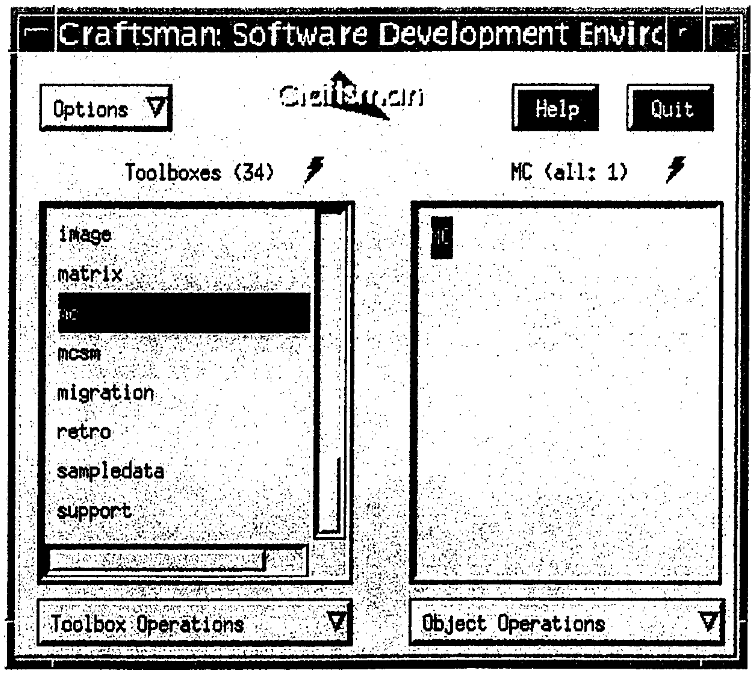
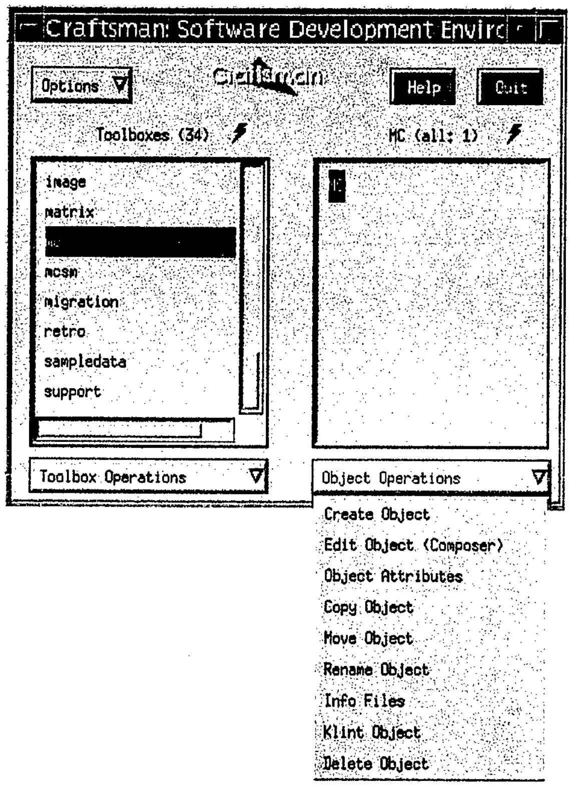


*Figure 4.5: Program Object Creation*

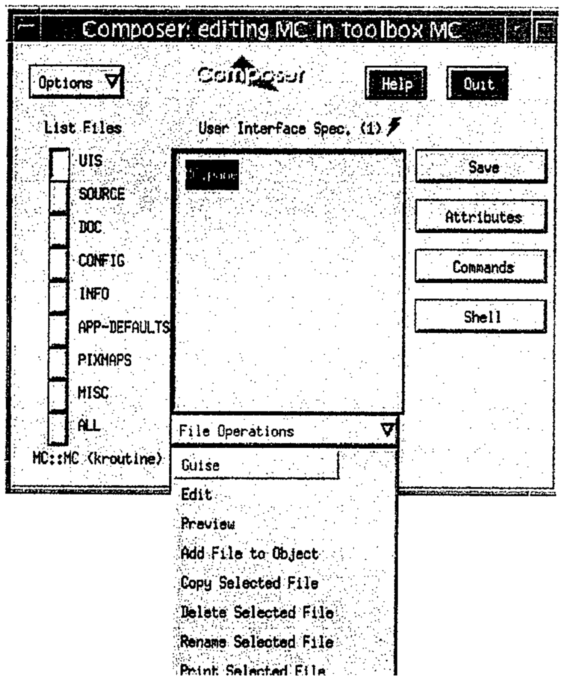
**Step 3. Create and Edit Graphical User Interface**

When **Craftsman** creates the **MC** program object, it copies a generic or template *User Interface Specification* (UIS) "MC.pane" file to the new program object directory. The "MC.pane" file specify the input and output arguments of the **MC** program. Since the user interface is not what we wanted, we need to edit the "MC.pane" using either a text editor or graphically using a KHOROS tool, the **Composer** application to provide the require input and output arguments.

(a) Select the **Edit Object (Composer)** from the **Object Operations** pulldown menu shown in Figure 4.6 (a).



(a)

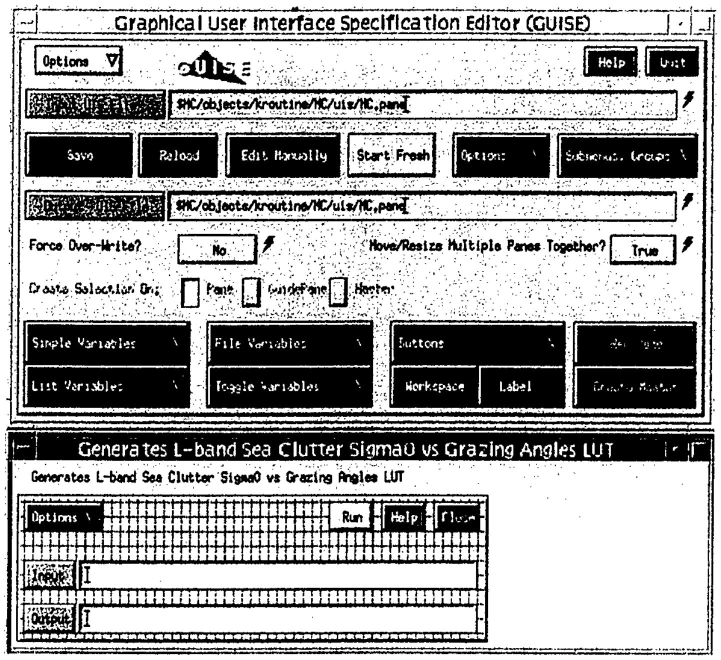


(b)

*Figure 4.6: (a) Edit Object (Composer) and (b) Composer Software Object Editor*

(b) This action invokes the **Composer** object editor shown in Figure 4.6(b). **Composer** editor allows editing and management of existing software objects. In addition, **Composer** allows GUI editing via **GUISE** (Graphical User Interface Specification Editor), sets software objects attributes, generate code, and compiles the program.

(c) We wish to modify the UIS file of the **MC** program interactively, so select *UIS* under the **List Files** (high-lighted by default), then select **Guise** from the **File Operations** drop-down menu as shown in Figure 4.6(b). **GUISE** is a graphical design tool used to create and modify user interfaces of software



$MC/objects/kroutine/MC/uis/MC.pane

$MC/objects/kroutine/MC/uis/MC.pane

Force Over-Write? No

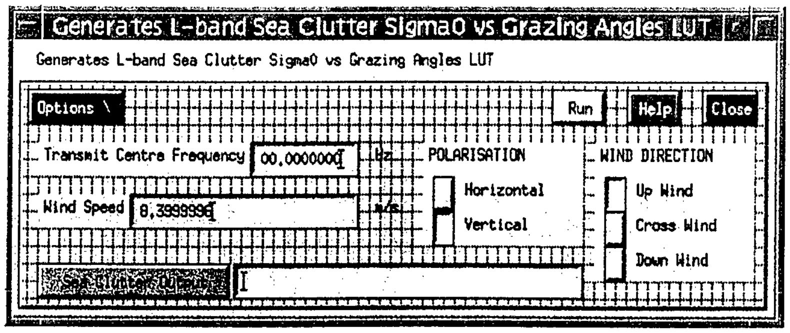
Move/Resize Multiple Panes Together? True

Create Selection On: Pane GuidePane Master

Generates L-band Sea Clutter SigmaO vs Grazing Angles LUT

*Figure 4.7: GUISE GUI Design Tool*

objects interactively. This action invokes two windows: (1) GUISE editing tools window; and (2) a standard GUI of the **MC** program is shown in Figure 4.7 (bottom figure) for further modification.



Generates L-band Sea Clutter SigmaO vs Grazing Angles LUT

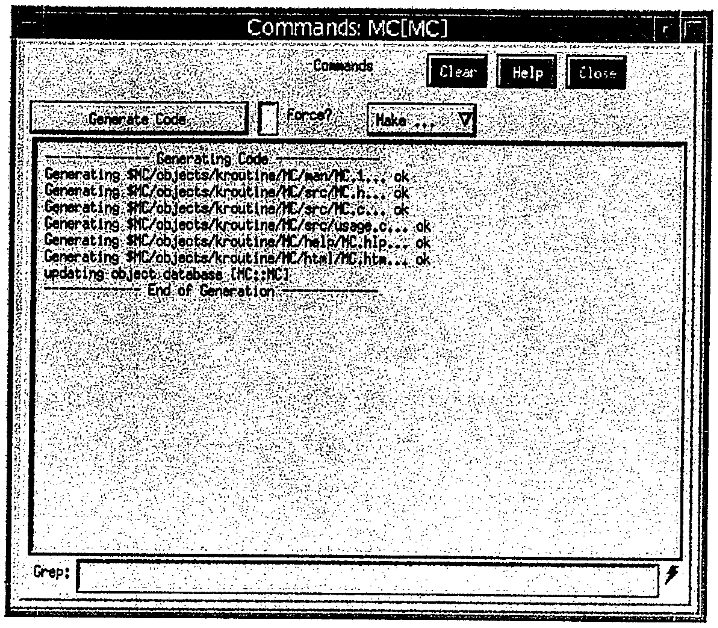
Transmit Centre Frequency 00.0000000 Hz

Wind Speed 8.3999996 m/s

*Figure 4.8:* **MC** *Sea Clutter GUI*

(d) After some editing, the final **MC** sea clutter GUI is shown in Figure 4.8.

(e) Whenever the UIS file or user interface of a program is modified, we have to regenerate the outdated KHOROS source code and documentation. Select **Commands** from the right side of **Composer** to invoke the **Commands** window. Then click the **Generate Code** button to update all source code and documentation based on the new user interface as specified in "MC.pane". While **Composer** is regenerating all source code and documentation for the program, it will display the regenerating status as shown in Figure 4.9.



```
———————— Generating Code ————————
Generating $MC/objects/kroutine/MC/man/MC.1... ok
Generating $MC/objects/kroutine/MC/src/MC.h... ok
Generating $MC/objects/kroutine/MC/src/MC.c... ok
Generating $MC/objects/kroutine/MC/src/usage.c... ok
Generating $MC/objects/kroutine/MC/help/MC.hlp... ok
Generating $MC/objects/kroutine/MC/html/MC.htm... ok
updating object database [MC::MC]
———————— End of Generation ————————
```

*Figure 4.9: Regenerating* **MC** *Sea Clutter GUI code and documentation*

### Step 4. Edit Source Code

At this stage, we can run the **MC** sea clutter glyph in **Cantata** minus the functionality. **Cantata** is a visual programming environment within the KHOROS system where programs are represented as icons or glyphs. Next, we must link the main KHOROS file, "MC.c", to our sea clutter program, "SeaClutter.c", and modify "SeaClutter.c" for KHOROS compliance.

(a) Insert the function prototype for sea clutter program,

```
void SeaClutter(void);
```

in the main KHOROS file "MC.c". Refer to Appendix A.1 for an example.

(b) Insert call to the sea clutter program in the main KHOROS file "MC.c"
In the main KHOROS source file labelled "MC.c", insert the call to the sea clutter program as follows,

```
/* main_library_call  */
```

SeaClutter();

```
/* main_library_call_end */
```

Refer to Appendix A.1 for an example.

(c) Insert the KHOROS header file,

```
#include "MC.h"
```

in "SeaClutter.c", so that the routine will understand KHOROS function calls.. An example of a KHOROS generated header file labelled "MC.h" can be found in Appendix A.2.

(d) Change main() in "SeaClutter.c" to a sub-routine (Refer to Appendix A.3 for an example).

(e) Change file declaration in "SeaClutter.c" to KHOROS file declaration (Refer to Appendix A.3 for an example).

(f) Comment or remove any default values. These values are now accessible through the **MC** GUI.

(g) Reassign KHOROS *Common Line User Interface* (CLUI) input variables to program variables. Note that CLUI variables are declared global variables and do not have to be passed during function calls.

(h) Change file I/O in "SeaClutter.c" to KHOROS file I/O.

(i) Edit the "Imakefile" and "Makefile" to include both the **MC** sea clutter source file "SeaClutter.c" and object file, "SeaClutter.o"

(j) After completing all the above editing, we need to recompile the source code and update all documentation. In the **Composer** window click on the **Commands** button and select the **Make Install** option. After compilation (without errors), the KHOROS **MC** sea clutter program can now be run in **Cantata**.

**Step 5. Documentation of the MC Sea Clutter Program**

Adequate documentation of newly created or updated programs is an important aspect of a software development process or life-cycle. The KHOROS system provides a framework for supporting and writing documentation.

During the creation of the **MC** software object, the KHOROS system automatically generates 3 types of documentation that conform to the KHOROS documentation standard: (1) an on-line manual or *man* page similar to the UNIX *man* page, describing the functionality and usage of the **MC** program; (2) an on-line help page accessible from within **Cantata**; and (3) a printed manual providing hardcopy presentation of the documentation.

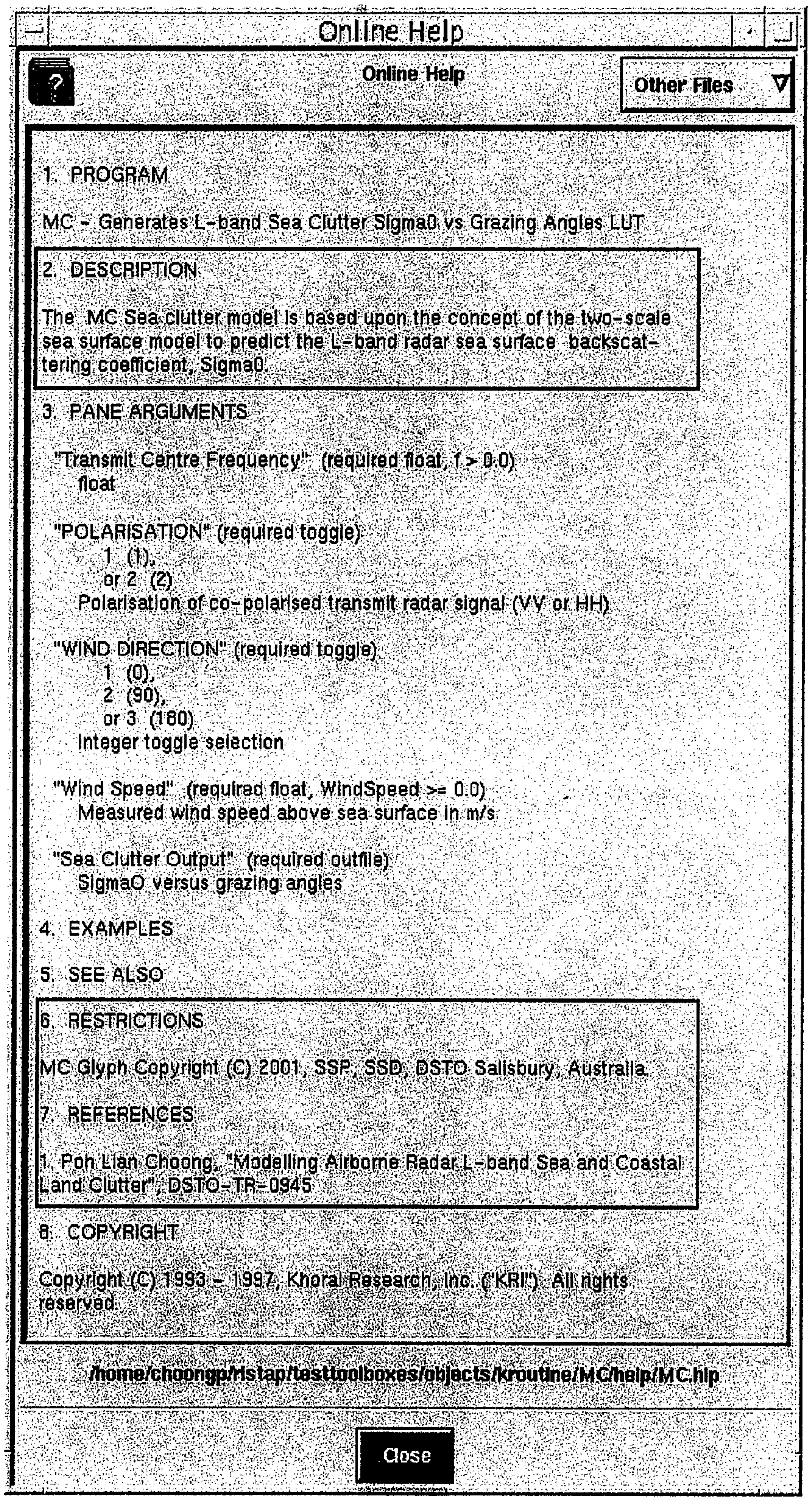
The above mentioned documentations can also be edited to provide additional descriptions and examples to the user. KHOROS has two categories of *man* pages: (1) one for programs ("MC.1" page); and (2) one for libraries ("MC.3" page).

(a) Editing **MC** Sea Clutter Program *Man* Page

All *man* pages are accessible from **Composer** by selecting the **DOC** files option. To edit the "MC.1" *man* page, select "MC.1" from the list, then select **Edit** from the **File Operations** pulldown list. This action invokes an editing window containing the "MC.1" *man* page with KHOROS tags that denote text segment. Edit the "MC.1" *man* page to add the desired documentation, then save the update upon completion.

(b) Updating **MC** Help Pages

To propagate the changes made in the "MC.1" *man* page to the help page, select the **Generate Code** from the **Composer Commands** button. You are able to view the newly formatted help page by selecting "MC.hlp" from the file list then select the **Formatted** option. Figure 4.10 shows an updated "MC.hlp" page with addition documentation highlighted in a box.



1. PROGRAM

MC - Generates L-band Sea Clutter Sigma0 vs Grazing Angles LUT

2. DESCRIPTION

The MC Sea clutter model is based upon the concept of the two-scale sea surface model to predict the L-band radar sea surface backscattering coefficient, Sigma0.

3. PANE ARGUMENTS

```
 "Transmit Centre Frequency"  (required float, f > 0.0)
   float

 "POLARISATION" (required toggle)
     1  (1),
     or 2  (2)
   Polarisation of co-polarised transmit radar signal (VV or HH)

 "WIND DIRECTION" (required toggle)
     1  (0),
     2  (90),
     or 3  (180)
   Integer toggle selection

 "Wind Speed"  (required float, WindSpeed >= 0.0)
   Measured wind speed above sea surface in m/s

 "Sea Clutter Output"  (required outfile)
   SigmaO versus grazing angles
```

4. EXAMPLES

5. SEE ALSO

6. RESTRICTIONS

MC Glyph Copyright (C) 2001, SSP, SSD, DSTO Salisbury, Australia.

7. REFERENCES

1. Poh Lian Choong, "Modelling Airborne Radar L-band Sea and Coastal Land Clutter", DSTO-TR-0945

8. COPYRIGHT

Copyright (C) 1993 - 1997, Khoral Research, Inc. ("KRI"). All rights reserved.

/home/choongp/Hstap/testtoolboxes/objects/kroutine/MC/help/MC.hlp

Figure 4.10: An Example of the MC Help Page

We have in this section describe a step-by-step approach to modifying and incorporating an existing stand-alone non-KHOROS program into the KHOROS environment. The process of generating *man* and help pages for associated KHOROS programs have also been briefly discussed. Figure 4.11 summarises the process of incorporating user stand-alone "C" code into the KHOROS environment.

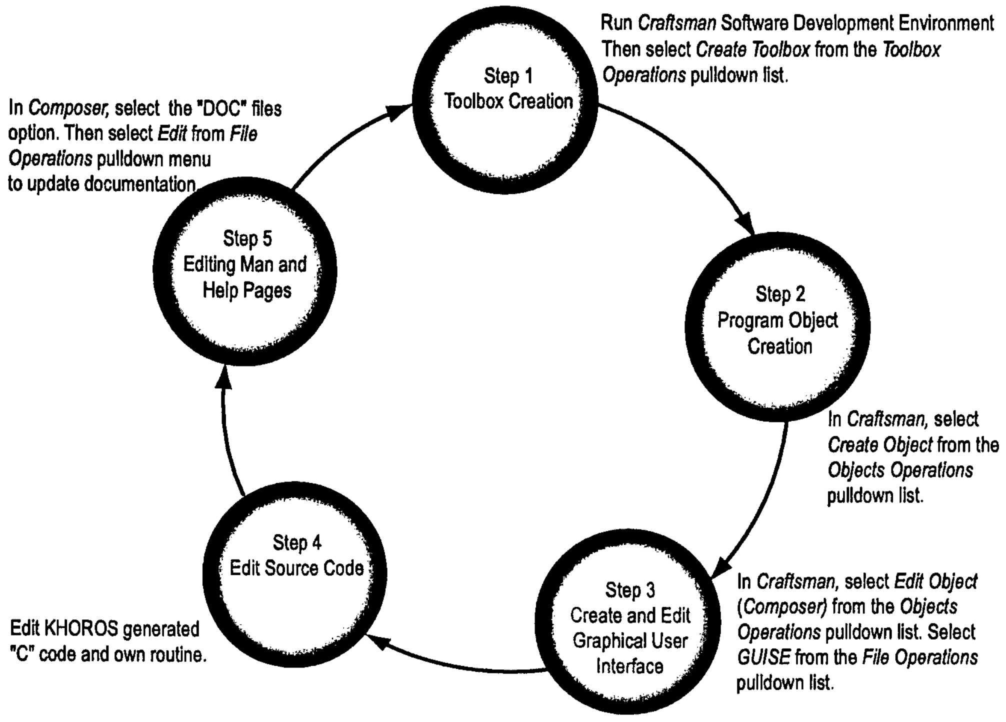

Step 1
Toolbox Creation
Run Craftsman Software Development Environment Then select Create Toolbox from the Toolbox Operations pulldown list.
Step 2
Program Object Creation
In Craftsman, select Create Object from the Objects Operations pulldown list.
Step 3
Create and Edit Graphical User Interface
In Craftsman, select Edit Object (Composer) from the Objects Operations pulldown list. Select GUISE from the File Operations pulldown list.
Step 4
Edit Source Code
Edit KHOROS generated "C" code and own routine.
Step 5
Editing Man and Help Pages
In Composer, select the "DOC" files option. Then select Edit from File Operations pulldown menu to update documentation

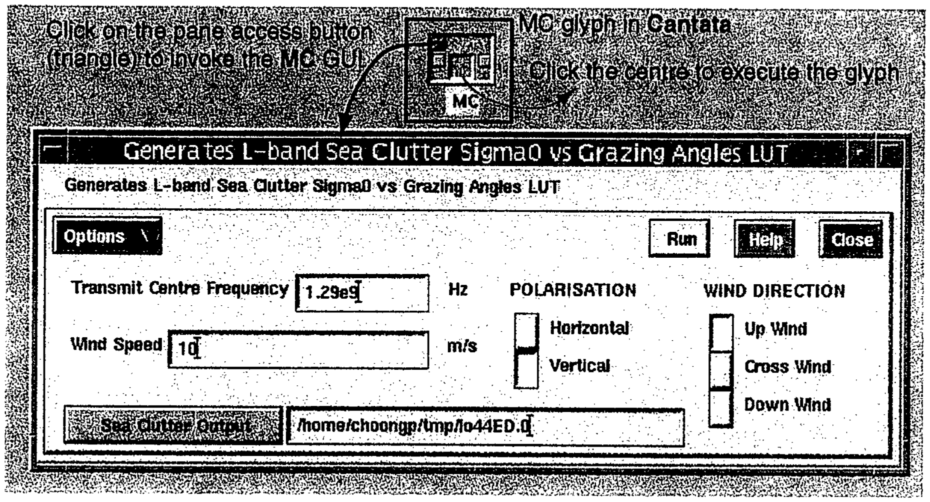
*Figure 4.11: Flow Diagram of the Glyph Creation Process*

# 5 Running the MC Sea Clutter Program in Cantata

Now that we have compiled, installed and edited the *man* and help pages for the **MC** sea clutter program, we can execute the **MC** program from within **Cantata**. Execute **Cantata** from the UNIX command line and wait for it to initialise. The **Cantata** workspace consists of a large canvas on which the program line-up can be constructed. The following steps illustrate a method for creating a program line-up in **Cantata** .

**Step 1. Selecting Glyph for Program Line-Up**

Select **Glyphs** from the selection bar at the top of the workspace. Then select **Sea Clutter** → **L-band Sea Clutter** → **MC** (category and sub-category attributes that we have previously inserted during the sea clutter toolbox creation process).



Generates L-band Sea Clutter Sigma0 vs Grazing Angles LUT

Generates L-band Sea Clutter Sigma0 vs Grazing Angles LUT

Transmit Centre Frequency 1.29e9 Hz

Wind Speed 10 m/s

Sea Clutter Output /home/choongp/tmp/lo44ED.0

*Figure 5.1:* **MC** *Glyph and GUI in* **Cantata**

**Step 2. Positioning the Selected Glyph onto the Cantata Workspace**

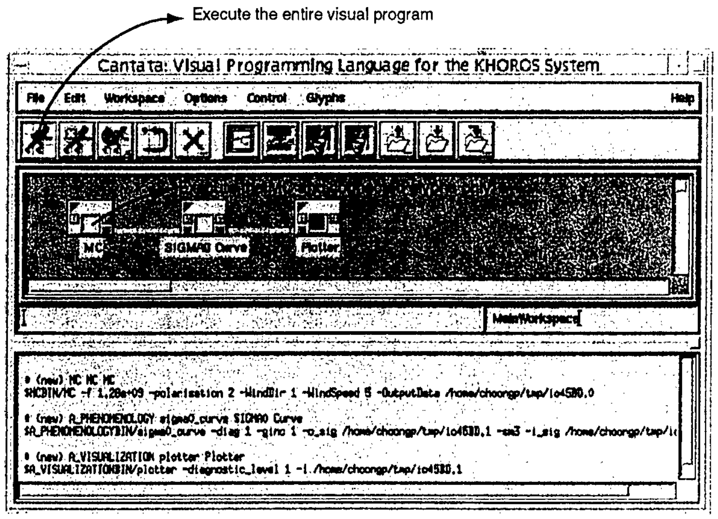
Highlight **MC** to select and drag the **MC** glyph to the **Cantata** workspace. Click once with the left mouse button to place the **MC** glyph at the desired position. You can reposition the glyph by selecting it with the left mouse button and moving it to the desired position. Release the mouse button to place the glyph in the new position.

**Step 3. Pane Information**

Figure 5.1 shows the **MC** glyph and if we click on the triangle located at the top-left side of the glyph (pane access button), we can view the **MC** GUI and all default values. These values can be updated before running the program.

Click on the **Help** button to access on-line information about the glyph functionality and variables descriptions. Figure 4.10 shows the "MC.hlp" page accessible via the **MC** glyph **Help** button.

A simple program line-up to plot the output of the **MC** glyph is shown in Figure 5.2. Click on either the run button located at the centre of the **MC** glyph to execute the line-up


Execute the entire visual program
Cantata: Visual Programming Language for the KHOROS System
File
Edit
Workspace
Options
Control
Glyphs
Help
MC
SIGMA0 Curve
Plotter
MainWorkspace

```
# (new) MC MC MC
$MCBIN/MC -f 1.28e+09 -polarisation 2 -WindDir 1 -WindSpeed 5 -OutputData /home/choongp/tmp/io4580.0

# (new) A_PHENOMENOLOGY sigma0_curve SIGMA0 Curve
$A_PHENOMENOLOGYBIN/sigma0_curve -diag 1 -gino 1 -o_sig /home/choongp/tmp/io4680.1 -sa3 -i_sig /home/choongp/tmp/i

# (new) A_VISUALIZATION plotter Plotter
$A_VISUALIZATIONBIN/plotter -diagnostic_level 1 -i /home/choongp/tmp/io4580.1
```

*Figure 5.2: Simple* **MC** *line-up*

or the **RUN** button from the selection bar at the top of the **Cantata** workspace. While executing, the simulation parameters are displayed in the bottom **Cantata** workspace as shown in Figure 5.2. An output plot of the backscatter coefficient, $\sigma^{\circ}$ versus grazing angle, $\psi$ is shown in Figure 5.3.

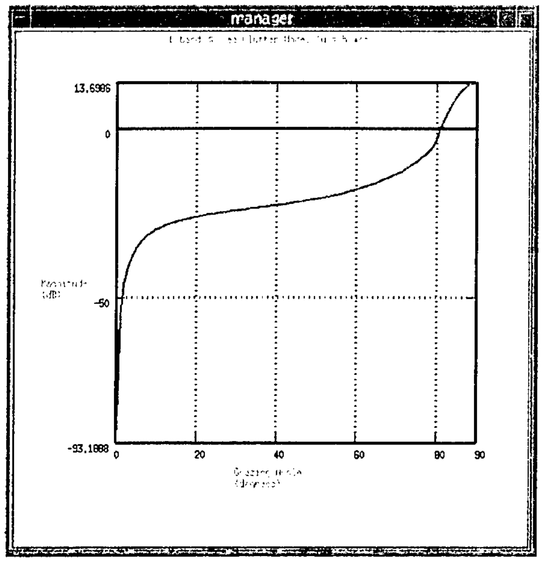


*Figure 5.3:* **MC** *Output plot*

## 5.1 A Comparison between the MC Sea Clutter Model and "RSS Sigma0 Curve" Model

We briefly compared the output of the **MC** sea clutter model and the "RSS Sigma0 Curve" Model for modelling the backscattering characteristics from the sea surface. We choose to compare these models for two sea conditions: (1) Low wind speed [wind velocity, U = 1.3 m/s (sea state 1)]; and (2) moderate wind speed with [wind velocity, U = 5 m/s (sea state 2)].

The plots shown in Figures 5.4 and 5.5 describe the $\sigma^\circ$ variation as $\psi$ varies from low to high values. The "RSS Sigma0 Curve" sea clutter model has a number of drawbacks: (1) the model does not account for specular returns for $\psi$ at near-normal incidence angles; and (2) the backscatter returns are not representative of measured L-band VV-polarised airborne radar data as shown in Figures 3.2 (a) - (b) and Figures 3.3 (a) - (c). .

# 6 Conclusions

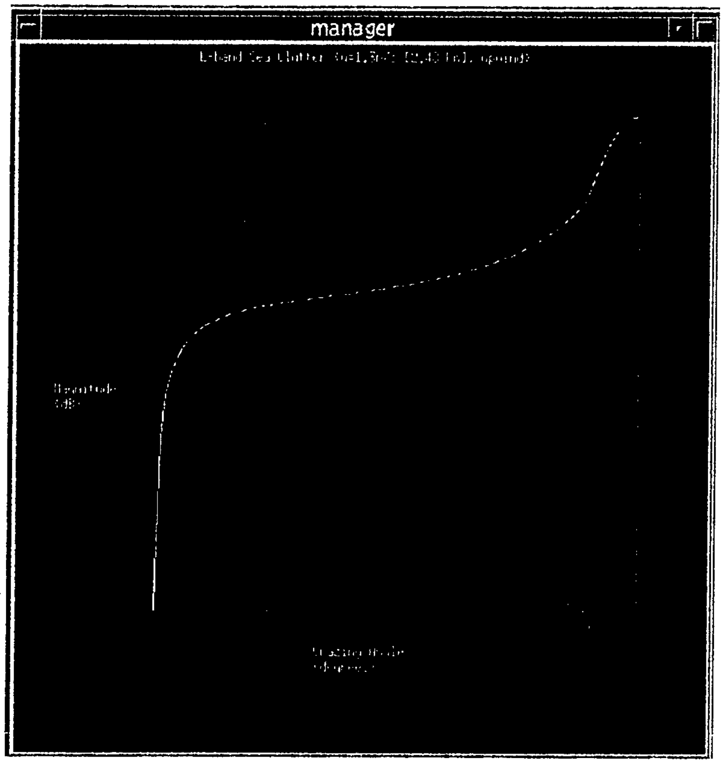
This report focused on two tasks: (1) the modelling of L-band VV-polarised sea clutter data; and (2) the ability to add functionality into the KHOROS environment for integration with RLSTAP programs.

In modelling L-band sea clutter data, the results of the comparisons have shown that the two-scale model modified for anisotropic sea spectrum, the **MC** sea clutter model, accurately predicted the NRL-4FR, JOSS I, and JOSS II data for a wide range of grazing angles and sea conditions. In addition, the **MC** sea clutter model accurately predicted the measured backscattering coefficient, $\sigma^\circ$, for the sea surface around the Port Darwin and Fannie Bay areas for grazing angles between 25° and 65°. However, this model still needs to be verified with backscattered signal acquired near normal incidence from sea temperature around 30°C.

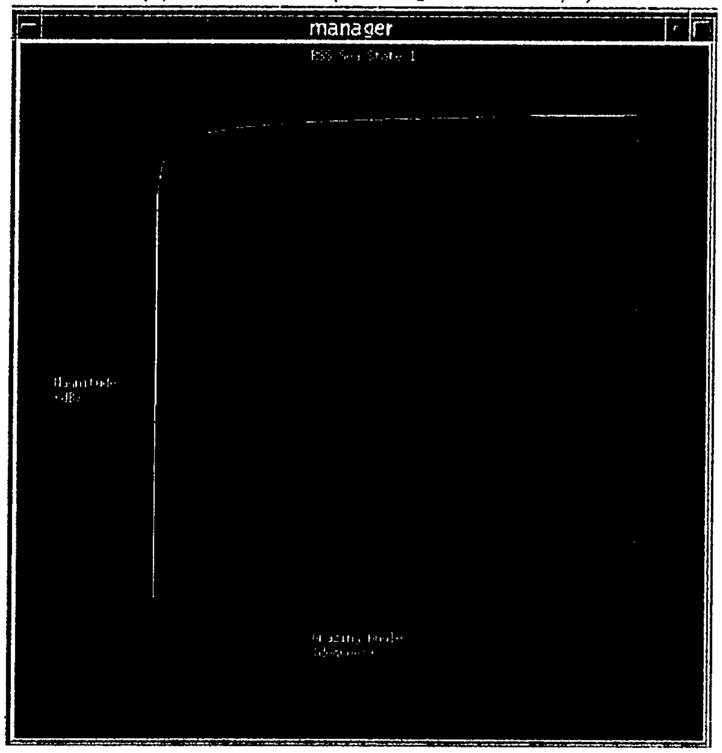
On the second task, we have modified an existing stand-alone L-band sea clutter program written in "C" for KHOROS compliance and integrated the model with existing **RLSTAP** programs with minimum code modification. This exercise has allowed us to examine the ability to add functionality and to further enhance **RLSTAP** modelling capabilities. The existing sea clutter program in **RLSTAP**, namely the "RSS Sigma0 Curve" program is inadequate for modelling L-band VV-polarised sea clutter data. The model is not representative of the measured NRL-4FR, JOSS I, JOSS II and Darwin POLSAR sea clutter data. The addition of the **MC** sea clutter glyph allows more accurate modelling of the microwave returns from the sea surface for radar operating in the L-band frequencies.

# Acknowledgements

The author would like to acknowledge the contributions of Drs. Nick Stacy, and Ed Kruzins (and team) for providing the POLSAR Darwin images and ground truth data.

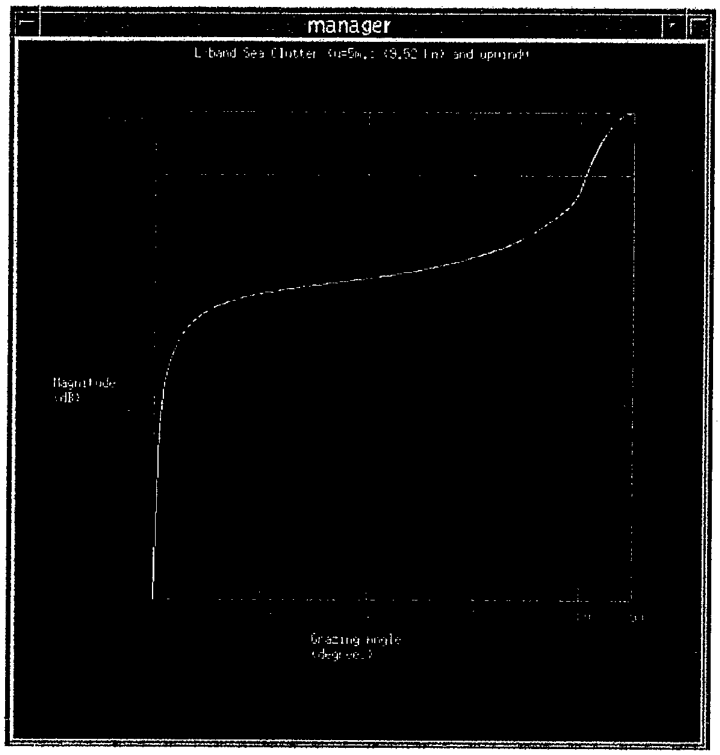


(a) **MC** Model (wind speed=1.3 m/s)

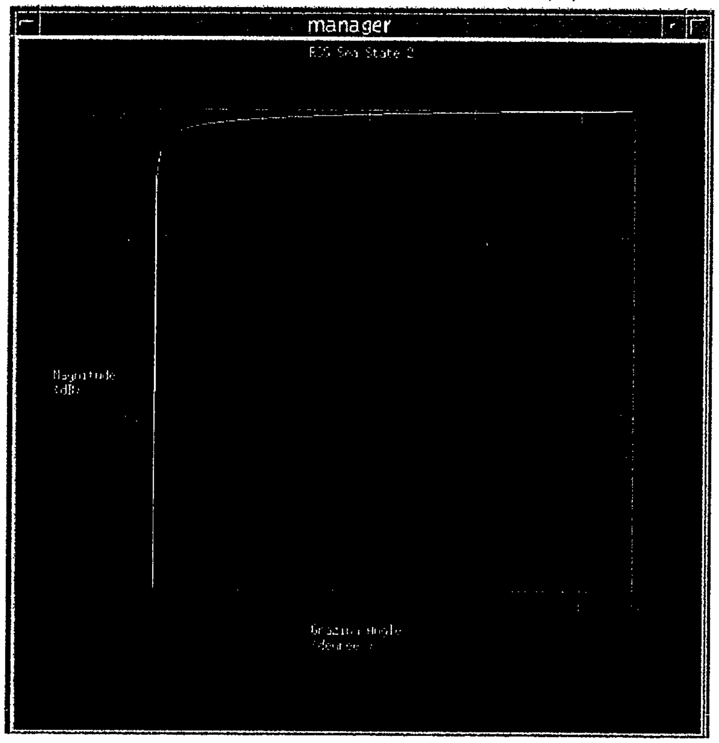


(b) RSS Sigma0 Curve (Sea State = 1)

*Figure 5.4: Comparison of* **MC** *and* RSS Sigma0 Curve *sea clutter modelling*



(a) **MC** Model (wind speed=5 m/s)



(b) RSS Sigma0 Curve (Sea State = 2)

*Figure 5.5: Comparison of* **MC** *and* RSS Sigma0 Curve *sea clutter modelling*

## References

1. G.R. Valenzuela, M.B. Laing, and J.C. Daley. Ocean Spectra for the High-frequency Waves as Determined from Airborne Radar Measurements. *J. Marine Res.*, 29(2):69–84, 1971.

2. N.W. Guinard and J.C. Daley. An Experimental Study of a Sea Clutter Model. *Proc. IEEE*, 58(4):543–550, April 1970.

3. J.C. Daley, W.T. Davis, and N.R. Mills. Radar Sea Return in High Sea States. Technical Report NRL Report 7142, Naval Research Laboratory, Wave Propagation Branch, Electronics Division, September 1970.

4. J.C. Daley, Jr. J.T. Ransone, and J.A. Burkett. Radar Sea Return - JOSS I. Technical Report NRL Report 7268, Naval Research Laboratory, Wave Propagation Branch, Electronics Division, May 1971.

5. K.S. Chen, A.K. Fung, and D.E. Weissman. A Backscattering Model for Ocean Surface. *IEEE Trans. Geosci. Remote Sensing*, 30(4):811–817, July 1992.

6. I. Antipov. Simulation of Sea Clutter Returns. Technical Report DSTO-TR-0679, Defence Science and Technology Organisation, Salisbury, June 1998.

7. L.B. Wetzel. *Radar Handbook*, chapter 13. Sea Clutter, pages 13.1–13.5. McGraw Hill, 2 edition, 1990.

8. J.C. Daley, Jr. J.T. Ransone, and W.T. Davis. Radar Sea Return - JOSS II. Technical Report NRL Report 7534, Naval Research Laboratory, Radar Division, February 1973.

9. M.A. Donelan and W.J. Pierson (JR). Radar Scattering and Equilibrium Ranges in Wind-Generated Waves With Application to Scatterometry. *J. Geophy. Res.*, 92(C5):4971–5029, May 1987.

10. G.R. Valenzuela. Theories for the Interaction of Electromagnetic and Oceanic Waves - A Review. *Boundary-Layer Meteorology*, 13:61–85, 1978.

11. G.R. Valenzuela and M.B. Laing. Study of Doppler Spectra of Radar Sea Echo. *J. Geophys. Res*, 75:551–563, January 1970.

12. E. Kordyban. Inserting C code into Khoros Visual Programming Environment. *TTCP SEN-TP3 Workshop, US AFRL Rome*, 2000.

13. Inc. Khoral Research. *KHOROS Toolbox Programming Manual: Introduction.* Khoral Research, Inc., 1997.

# Appendix A: KHOROS and Non-KHOROS Source Files

## A.1 Main Program for KHOROS Generated MC.c Source File

```
 /*
  * Khoros: $Id$
  */

#if !defined(__lint) && !defined(__CODECENTER__)
static char rcsid[] = "Khoros: $Id$";
#endif

/
 * Copyright (C) 1993 - 1997, Khoral Research, Inc., ("KRI").
 * All rights reserved.  See $BOOTSTRAP/repos/license/License or run klicense.
 */

/* >>>>>>>>>>>>>>>>>>>>>>>>>>>>>> <<<<<<<<<<<<<<<<<<<<<<<<<<
   >>>>
   >>>>  Main program for MC
   >>>>
   >>>>  Private:
   >>>>  main
   >>>>
   >>>>   Static:
   >>>>   Public:
   >>>>
   >>>>>>>>>>>>>>>>>>>>>>>>>>>>>> <<<<<<<<<<<<<<<<<<<<<<<<<< */


#include "MC.h"

clui_info_struct *clui_info = NULL;

void SeaClutter(void); /* ####### (a) INSERT MC FUNCTION PROTOTYPE ########### */
/*-----------------------------------------------------------
|
| Routine Name: main() - Generates L-band Sea Clutter Sigma0 vs Grazing Angles LUT
|
| Purpose: main program for MC
|
| Input:
| float clui_info->f_float; {float}
| int   clui_info->f_flag; {TRUE if -f specified}
|
| int clui_info->polarisation_toggle; {Polarisation of co-polarised transmit radar signal (VV or HH)}
| int clui_info->polarisation_flag; {TRUE if -polarisation specified}
|
| int clui_info->WindDir_toggle; {Integer toggle selection}
| int clui_info->WindDir_flag; {TRUE if -WindDir specified}
|
| float clui_info->WindSpeed_float; {Measured wind speed above sea surface in m/s}
| int   clui_info->WindSpeed_flag; {TRUE if -WindSpeed specified}
|
| char *clui_info->OutputData_file; {Sigma0 versus grazing angles}
| int   clui_info->OutputData_flag; {TRUE if -OutputData specified}
|
| Output:
| Returns:
|
| Written By: Poh Lian Choong
| Date:
| Modifications:
|
------------------------------------------------------------*/

int main(
   int  argc,
   char **argv)
{

/* -main_variable_list */
/* -main_variable_list_end */

khoros_init(argc, argv, "MC", PRODUCT_RELEASE_DATE,
            PRODUCT_RELEASE_NAME, PRODUCT_RELEASE_VERSION,
            PRODUCT_RELEASE_MAJOR, PRODUCT_RELEASE_MINOR,
            "$MC/objects/kroutine/MC");
kexit_handler(MC_free_args, NULL);

/* -main_get_args_call */
kclui_init("MC", "MC", KGEN_KROUTINE,  &clui_uis_spec,
   MC_usage_additions, MC_get_args,
     MC_free_args);
/* -main_get_args_call_end */

/* -main_before_lib_call */
/* -main_before_lib_call_end */

/* -main_library_call */
SeaClutter(); /* ## (b) INSERT CALL TO L-band SEA CLUTTER PROGRAM/FUNCTION ###*/
/* -main_library_call_end */

/* -main_after_lib_call */
/* -main_after_lib_call_end */


kexit(KEXIT_SUCCESS);
}


/*-----------------------------------------------------------
|
|  Routine Name: MC_usage_additions
|
|       Purpose: Prints usage additions in MC_usage routine
|
|         Input: None
|
|        Output: None
|    Written By: ghostwriter -oname MC
|          Date: July 1, 1997
| Modifications:
|
------------------------------------------------------------*/
void MC_usage_additions(void)
{
kfprintf(kstderr, "\tGenerates L-band Sea Clutter Sigma0 vs Grazing Angles LUT\n");

/* -usage_additions */
/* -usage_additions_end */

}
/*-----------------------------------------------------------
|
|  Routine Name: MC_free_args
|
|       Purpose: Frees CLUI struct allocated in MC_get_args()
|
|         Input: None
|
|        Output: None
|    Written By: ghostwriter -oname MC
|          Date: July 1, 1997
| Modifications:
|
------------------------------------------------------------*/
/* ARGSUSED */
void
MC_free_args(
    kexit_status status,
    kaddr        client_data)
{

/* do the wild and free thing */
if (clui_info != NULL)
{
kfree_and_NULL(clui_info->OutputData_file);
kfree_and_NULL(clui_info);
}

/* -free_handler_additions */
/* -free_handler_additions_end */
}
```

## A.2 KHOROS Generated Header File - MC.h

```
 /*
  * Khoros: $Id$
  */

/*
 * Copyright (C) 1993 - 1997, Khoral Research, Inc., ("KRI").
 * All rights reserved.  See $BOOTSTRAP/repos/license/License or run klicense.
 */

/* >>>>>>>>>>>>>>>>>>>>>>>>>>>>> <<<<<<<<<<<<<<<<<<<<<<<<<<
   >>>>
   >>>>       Purpose: Include file for MC
   >>>>
   >>>>    Written By:
   >>>>
   >>>> Modifications:
   >>>>
   >>>>          Date: July 1, 1997
   >>>>
   >>>>>>>>>>>>>>>>>>>>>>>>>>>>>> <<<<<<<<<<<<<<<<<<<<<<<<<< */


#ifndef _MC_h_
#define _MC_h_

        /*-------------------------------------*
         |       #includes
        ---------------------------------------*/

#include <mc.h>

/* -include_includes */
/* -include_includes_end */


        /*-------------------------------------*
         |       #defines
        ---------------------------------------*/

/* -include_defines */
/* -include_defines_end */

        /*-------------------------------------*
         |       typedefs
        ---------------------------------------*/

typedef struct _clui_info_struct {

/*
 *  float (required float)
 */
float  f_float; /* float FLOAT */
int  f_flag;    /* float FLAG */

/*
 * Polarisation of co-polarised transmit radar signal (VV or HH) (Required integer toggle)
 * 1 (1)
 * 2 (2),
 */
int  polarisation_toggle; /* Polarisation of co-polarised transmit radar signal (VV or HH) INT TOGGLE */
int  polarisation_flag; /* Polarisation of co-polarised transmit radar signal (VV or HH) FLAG */

/*
 * Integer toggle selection (Required integer toggle)
 * 1 (0)
 * 2 (90)
 * 3 (180),
 */
int  WindDir_toggle; /* Integer toggle selection INT TOGGLE */
int  WindDir_flag; /* Integer toggle selection FLAG */

/*
 *  Measured wind speed above sea surface in m/s (required float)
 */
float  WindSpeed_float; /* Measured wind speed above sea surface in m/s FLOAT */
int  WindSpeed_flag;    /* Measured wind speed above sea surface in m/s FLAG */

/*
 *  Sigma0 versus grazing angles (required outfile)
 */
char *OutputData_file; /* Sigma0 versus grazing angles FILENAME */
int  OutputData_flag; /* Sigma0 versus grazing angles FLAG */

} clui_info_struct;

/* -include_typedefs */
/* -include_typedefs_end */


        /*-------------------------------------*
         |      global variable declarations
        --------------------------------------*/

extern clui_info_struct *clui_info;
extern uis_struct clui_uis_spec;
/* -include_variables */
/* -include_variables_end */


        /*-------------------------------------*
         |      macros
        --------------------------------------*/

/* -include_macros */
/* -include_macros_end */

        /*-------------------------------------*
         |      routine definitions
        --------------------------------------*/

int main PROTO((int, char **));
void MC_get_args PROTO((kaddr));
void MC_usage_additions PROTO((void));
void MC_free_args PROTO((kexit_status, kaddr));
```

```
/* -include_routines */
/* -include_routines_end */

#endif
```

## A.3 Main MC Sea Clutter Program - "SeaClutter.c"

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "MC.h"                    /* #########  (c) INSERT KHOROS HEADER FILE ###########*/
#ifndef _FCOMPLEX_DECLARE_T_
typedef struct FCOMPLEX {float r,i;} fcomplex;
#define _FCOMPLEX_DECLARE_T_
#endif /* _FCOMPLEX_DECLARE_T_ */

static float sqrarg;
#define SQR(a) ((sqrarg=(a)) == 0.0 ? 0.0 : sqrarg*sqrarg)

static float maxarg1,maxarg2;
#define FMAX(a,b) (maxarg1=(a),maxarg2=(b),(maxarg1) > (maxarg2) ?\
        (maxarg1) : (maxarg2))

#define FREERETURN {free_vector(fvec,1,n);free_vector(xold,1,n);\
        free_vector(p,1,n);free_vector(g,1,n);free_matrix(fjac,1,n,1,n);\
        free_ivector(indx,1,n);return;}

int nn;
float *fvec,w,z;

#define NR_END 1
#define FREE_ARG char*
#define HORIZONTAL 1
#define VERTICAL 2
#define UPWIND 0
#define DOWNWIND 180
#define CROSSWIND 90
#define Z 1250
#define Km2 13.177e4;                 /* sea surface tension and density */
#define E1 72  /* L-band sea refractivity real value */
#define E2 -59  /* L-band sea refractivity complex value */
#define NAngle 1000
#define psiMin 0.1
#define psiMax 89.9
#define psiI 0.1
#define SPECULAR 60;
#define pi 3.14159265
#define MAXITS 200
#define TOLF 1.0e-4
#define TOLMIN 1.0e-6
#define TOLX 1.0e-7
#define STPMX 100.0
#define N 1
#define ALF 1.0e-4
#define TOLX 1.0e-7
#define TINY 1.0e-20
#define EPS 1.0e-4

/*---------------------------------------- Main Files ---------------------------------------- */
void SeaClutter(void);
void (*nrfuncv)(int , float *, float *);
void funcv(int, float *,float *);
void newt(float *, int , int *,void (*vecfunc)(int, float [], float []));
```

```
/*--------------------------------------- Utilities Files ------------------------------------ */
fcomplex Complex(float,float);
fcomplex Cadd(fcomplex, fcomplex);
fcomplex Csub(fcomplex, fcomplex);
fcomplex Cmul(fcomplex, fcomplex);
fcomplex Cdiv(fcomplex, fcomplex);
float    Cabs(fcomplex);
fcomplex Csqrt(fcomplex);
void     nrerror(char *);
float    *vector(long,long);
int      *ivector(long,long)
fcomplex *cvector(long,long);
float    **matrix(long,long,long,long);
void     free_vector(float *,long,long);
void     free_ivector(int *,long,long);
void     free_cvector(fcomplex *,long,long);
void     free_matrix(float **,long,long,long,long);


/*--------------------------------------------------------------------------------------------
   void SeaClutter(void)  /* ############# (d) CHANGE main() to a SUBROUTINE ############ */
   Calculate the L-band backscattering sea slutter returns using the MC Sea Clutter model
   (refer to report DSTO-TR-0945)

   Copyright: Poh Lian Choong@DSTO Salisbury, Australia, 27 February 2001
   Version 1.0 05 March 2001 modified for RLSTAP compliance
  --------------------------------------------------------------------------------------------*/
void SeaClutter(void)
{
 float    *psi,*psiS,*A,k,*K,*K2,v,V,*U,*f,p,W1,*W2,*sigmaB,*sigmaS,*sigma,R2,s2;
 float    tmp,tmp1,tmp2,tmp3;
 float    u,freq;
 int      pol, l;
 int      i,index,check;
 fcomplex Er,*R,*P,*Q,S,Ctmp,a,b;
 FILE     *outputfile;
 kfile    *output_file;     /* #### (e) CHANGE FILE DECLARATION to KHOROS FILE DECLARATION #### */

/* ### (g) REASSIGN KHOROS CLUI VARIABLES to PROGRAM VARIABLES ########### */
 freq     = clui_info->f_float;
 pol      = clui_info->polarisation_toggle;
 u        = clui_info->WindSpeed_float;
 l        = clui_info->WindDir_toggle;

 psi      = vector(1,NAngle);
 psiS     = vector(1,NAngle);
 R        = cvector(1,NAngle);
 P        = cvector(1,NAngle);
 Q        = cvector(1,NAngle);
 A        = vector(1,NAngle);
 K        = vector(1,NAngle);
 K2       = vector(1,NAngle);
 W2       = vector(1,NAngle);
 sigmaB   = vector(1,NAngle);
 sigmaS   = vector(1,NAngle);
 sigma    = vector(1,NAngle);
 U        = vector(1,2);
```

```
f       = vector(1,2);

for (i=1;i<(psiMax*10+1);i++) *(psi+i)=psiI*i*pi/180;

/* Determine L-band sea reflection coefficient */
Er = Complex(E1,E2);
for (i=1;i<(psiMax*10+1);i++){
  Ctmp   = Complex(cos(*(psi+i))*cos(*(psi+i)),0);
  *(R+i) = Csqrt(Csub(Er,Ctmp));
  *(P+i) = Complex(sin(*(psi+i)),0);
  *(Q+i) = Cmul(Complex(cos(*(psi+i)),0),Complex(cos(*(psi+i)),0)); /* Q^2 */
}
S =Csub(Er,Complex(1,0));

switch(pol){
 case HORIZONTAL:
      for (i=1;i<(psiMax*10+1);i++){
       Ctmp   = Cadd(*(P+i),*(R+i));
       b      = Cmul(Ctmp,Ctmp);
       *(A+i) = SQR(Cabs(Cdiv(S,b)));
      }
      break;

 case VERTICAL:
      for (i=1;i<(psiMax*10+1);i++){
       Ctmp   = Cmul(Er,Cadd(*(Q+i),Complex(1,0)));
       a      = Cmul(S,Csub(Ctmp,*(Q+i)));
       Ctmp   = Cadd(Cmul(Er,*(P+i)),*(R+i));
       b      = Cmul(Ctmp,Ctmp);
       *(A+i) = SQR(Cabs(Cdiv(a,b)));
      }
      break;
}

free_cvector(Q,1,NAngle);
free_cvector(P,1,NAngle);
free_cvector(R,1,NAngle);

/* --------- Determine sea surface 2-D directional sea spectrum --------- */
k = (2*pi*freq)/3e8;                /* radar wave number, k=2*pi/lambda */
for (i=1;i<(psiMax*10+1);i++){
 *(K+i) = 2*k*cos(*(psi+i));        /* Braggs wave number (rad/m) */
 *(K2+i)= SQR(*(K+i))/Km2;          /* Squared ocean wave number */
}
l = l*(pi/180);                     /* convert wind direction to radian */

if (u >= 5) v = (0.003+0.84e-3*u)/(0.005+0.78e-3*u);
else v = (0.003+1.92e-3*u)/(3.16e-3*u); /* ratio of slope variance (clear) */

V = 2*(1-v)/(1+v);

U[1]=1.0; z = Z, w=u;
newt(U,N,&check,funcv);             /* Determine friction velocity */
funcv(N,U,f);
p = 5.0 - log10(U[1]);
free_vector(U,1,2);
free_vector(f,1,2);
```

```
for (i=1;i<(psiMax*10+1);i++){
 tmp2= *(K+i); tmp3 = *(K2+i);
 tmp = (1.0+3.0*tmp3)*pow(9.81,(1-p)/2);
 tmp1= pow(tmp2,0.922)*pow(tmp2*(1+tmp3),(1+p)/2);
 if (u<5) W1 = 1.750*pow(2*pi,p-1)*(tmp/tmp1);
 else W1 = 0.875*pow(2*pi,p-1)*(tmp/tmp1);
 *(W2+i) = (1/(2*pi))*(1+V*cos(2*l))*W1;
}

/* --------- Determine Bragg's component --------- */

for (i=1;i<(psiMax*10+1);i++){
 tmp = SQR(k)*pow(sin(*(psi+i)),4);
 *(sigmaB+i) = 10*log10(tmp*(*(A+i))*(*(W2+i)));
 *(sigma+i)  = *(sigmaB+i);
}

/* --------- Determine specular components --------- */
a = Cmul(Complex(0.52,0),Csub(Er,Complex(1,0)));
b = Cmul(Cadd(Csqrt(Er),Complex(1,0)),Cadd(Csqrt(Er),Complex(1,0)));
R2 = SQR(Cabs(Cdiv(a,b)));

if (u>=5) s2 = 1.56e-3*u;
else s2 = 5.12e-3*u;

for(i=601;i<(psiMax*10+1);i++) {
 *(psiS+i)=(90.0 - i*psiI)*pi/180;
 tmp1 = (R2/s2)*pow(1/cos(*(psiS+i)),4);
 tmp2 = exp(-SQR(tan(*(psiS+i)))/s2);
 *(sigmaS+i)=10*log10(tmp1*tmp2);
}

/* --------- Determine the intersection of Bragg and specular components --------- */
for(i=601;i<(psiMax*10+1);i++) {
 if (*(sigmaS+i) >=*(sigmaB+i)) {
  index = i;
   break;
 }
}

for(i=(index+1);i<(psiMax*10+1);i++) *(sigma+i)=*(sigmaS+i);

/* ########### (h) Change "C" FILE  I/O to KHOROS FILE I/O ############## */
output_file=kfopen(clui_info->OutputData_file,"w");
for(i=1;i<=index;i++)
  kfprintf(output_file,"%f %f\n",i*psiI,*(sigma+i));
for(i=(index+1);i<(psiMax*10+1);i++)
  kfprintf(output_file,"%f %f\n",i*psiI,*(sigma+i));
kfclose(output_file);

free_vector(sigma,1,NAngle);
free_vector(sigmaS,1,NAngle);
free_vector(sigmaB,1,NAngle);
free_vector(W2,1,NAngle);
free_vector(K2,1,NAngle);
free_vector(K,1,NAngle);
free_vector(A,1,NAngle);
free_vector(psiS,1,NAngle);
```

```
 free_vector(psi,1,NAngle);

}/* SeaClutter() */

/*--------------------------------------------------------------
  OBJECTIVE FUNCTIONS
  -------------------------------------------------------------- */
void funcv(int n, float x[],float f[])
{
 float L;

 L = (0.684/x[1])+4.28e-5*SQR(x[1])-0.0443;
 f[1] = (x[1]/0.4)*log(z/L)*1e-2-w;
}

/*----------------------------------------------------------------------------
 OPTIMISATION ROUTINES (from Numerical Recipes in C)
 ---------------------------------------------------------------------------- */
void newt(float x[], int n, int *check,void (*vecfunc)(int, float [], float []))
{
 void fdjac(int n, float x[], float fvec[], float **df,void (*vecfunc)(int, float [], float []));
 float fmin(float x[]);
 void lnsrch(int n, float xold[],float fold,float g[],float p[],float x[],float
 *f,float stpmax, int *check, float (*func)(float []));
 void lubksb(float **a, int n, int *indx, float b[]);
 void ludcmp(float **a, int n, int *indx, float *d);


 int i,its,j,*indx;
 float d,den,f,fold,stpmax,sum,temp,test,**fjac,*g,*p,*xold;

  indx=ivector(1,n);
  fjac=matrix(1,n,1,n);
  g=vector(1,n);
  p=vector(1,n);
  xold=vector(1,n);
  fvec=vector(1,n);

  nn=n;

nrfuncv=vecfunc;
f=fmin(x);
test=0.0;
for (i=1;i<=n;i++)
        if (fabs(fvec[i]) > test) test=fabs(fvec[i]);
        if (test < 0.01*TOLF) {
                *check=0;
                FREERETURN
        }
        for (sum=0.0,i=1;i<=n;i++) sum += SQR(x[i]);
        stpmax=STPMX*FMAX(sqrt(sum),(float)n);
        for (its=1;its<=MAXITS;its++) {
                fdjac(n,x,fvec,fjac,vecfunc);
                for (i=1;i<=n;i++) {
                        for (sum=0.0,j=1;j<=n;j++) sum += fjac[j][i]*fvec[j];
                        g[i]=sum;
                }
                for (i=1;i<=n;i++) xold[i]=x[i];
                fold=f;
                for (i=1;i<=n;i++) p[i] = -fvec[i];
                ludcmp(fjac,n,indx,&d);
                lubksb(fjac,n,indx,p);
                lnsrch(n,xold,fold,g,p,x,&f,stpmax,check,fmin);
                test=0.0;
                for (i=1;i<=n;i++)
                        if (fabs(fvec[i]) > test) test=fabs(fvec[i]);
                if (test < TOLF) {
                        *check=0;
                        FREERETURN
                }
                if (*check) {
                        test=0.0;
                        den=FMAX(f,0.5*n);
                        for (i=1;i<=n;i++) {
                                temp=fabs(g[i])*FMAX(fabs(x[i]),1.0)/den;
                                if (temp > test) test=temp;
                        }
                        *check=(test < TOLMIN ? 1 : 0);
                        FREERETURN
                }
                test=0.0;
                for (i=1;i<=n;i++) {
                        temp=(fabs(x[i]-xold[i]))/FMAX(fabs(x[i]),1.0);
                        if (temp > test) test=temp;
                }
                if (test < TOLX) FREERETURN
        }
        nrerror("MAXITS exceeded in newt");
}
/*-----------------------------------------------------------------------------
  OPTIMIZATION UTILITIES (from Numerical Recipes in C)
  ----------------------------------------------------------------------------- */
void lnsrch(int n, float xold[], float fold, float g[], float p[], float x[],
                    float *f, float stpmax, int *check, float (*func)(float []))
{
 int i;
 float a,alam,alam2,alamin,b,disc,f2,fold2,rhs1,rhs2,slope,sum,temp, test,tmplam;
 *check=0;
 for (sum=0.0,i=1;i<=n;i++) sum += p[i]*p[i];
  sum=sqrt(sum);
  if (sum > stpmax)
   for (i=1;i<=n;i++) p[i] *= stpmax/sum;
   for (slope=0.0,i=1;i<=n;i++) slope += g[i]*p[i];
   test=0.0;
   for (i=1;i<=n;i++) {
    temp=fabs(p[i])/FMAX(fabs(xold[i]),1.0);
    if (temp > test) test=temp;
   }
   alamin=TOLX/test;
   alam=1.0;
   for (;;) {
    for (i=1;i<=n;i++) x[i]=xold[i]+alam*p[i];
     *f=(*func)(x);
     if (alam < alamin) {
      for (i=1;i<=n;i++) x[i]=xold[i];
       *check=1;
```

```
        return;
       } else if (*f <= fold+ALF*alam*slope) return;
       else {
        if (alam == 1.0)
         tmplam = -slope/(2.0*(*f-fold-slope));
        else {
         rhs1 = *f-fold-alam*slope;
         rhs2=f2-fold2-alam2*slope;
         a=(rhs1/(alam*alam)-rhs2/(alam2*alam2))/(alam-alam2);
         b=(-alam2*rhs1/(alam*alam)+alam*rhs2/(alam2*alam2))/(alam-alam2);
         if (a == 0.0) tmplam = -slope/(2.0*b);
         else {
          disc=b*b-3.0*a*slope;
          if (disc<0.0) nrerror("Roundoff problem in lnsrch.");
          else tmplam=(-b+sqrt(disc))/(3.0*a);
         }
         if (tmplam>0.5*alam) tmplam=0.5*alam;
    }
   }
   alam2=alam;
   f2 = *f;
   fold2=fold;
   alam=FMAX(tmplam,0.1*alam);
 }
} /* lnsrch() */

float fmin(float x[])
{
        int i;
        float sum;

        (*nrfuncv)(nn,x,fvec);
        for (sum=0.0,i=1;i<=nn;i++) sum += SQR(fvec[i]);
        return 0.5*sum;
} /* fmin() */

void lubksb(float **a, int n, int *indx, float b[])
{
        int i,ii=0,ip,j;
        float sum;

        for (i=1;i<=n;i++) {
                ip=indx[i];
                sum=b[ip];
                b[ip]=b[i];
                if (ii)
                        for (j=ii;j<=i-1;j++) sum -= a[i][j]*b[j];
                else if (sum) ii=i;
                b[i]=sum;
        }
        for (i=n;i>=1;i--) {
                sum=b[i];
                for (j=i+1;j<=n;j++) sum -= a[i][j]*b[j];
                b[i]=sum/a[i][i];
        }
}/* lubksb() */

void ludcmp(float **a, int n, int *indx, float *d)
```

```
{
        int i,imax,j,k;
        float big,dum,sum,temp;
        float *vv;

        vv=vector(1,n);
        *d=1.0;
        for (i=1;i<=n;i++) {
                big=0.0;
                for (j=1;j<=n;j++)
                        if ((temp=fabs(a[i][j])) > big) big=temp;
                if (big == 0.0) nrerror("Singular matrix in routine ludcmp");
                vv[i]=1.0/big;
        }
        for (j=1;j<=n;j++) {
                for (i=1;i<j;i++) {
                        sum=a[i][j];
                        for (k=1;k<i;k++) sum -= a[i][k]*a[k][j];
                        a[i][j]=sum;
                }
                big=0.0;
                for (i=j;i<=n;i++) {
                        sum=a[i][j];
                        for (k=1;k<j;k++)
                                sum -= a[i][k]*a[k][j];
                        a[i][j]=sum;
                        if ( (dum=vv[i]*fabs(sum)) >= big) {
                                big=dum;
                                imax=i;
                        }
                }
                if (j != imax) {
                        for (k=1;k<=n;k++) {
                               dum=a[imax][k];
                                a[imax][k]=a[j][k];
                                a[j][k]=dum;
                        }
                        *d = -(*d);
                        vv[imax]=vv[j];
                }
                indx[j]=imax;
                if (a[j][j] == 0.0) a[j][j]=TINY;
                if (j != n) {
                        dum=1.0/(a[j][j]);
                        for (i=j+1;i<=n;i++) a[i][j] *= dum;
                }
        }
        free_vector(vv,1,n);
}/* ludcmp() */

void fdjac(int n,float x[],float fvec[],float **df,void (*vecfunc)(int,float []
,float []))
{
        int i,j;
        float h,temp,*f;

        f=vector(1,n);
        for (j=1;j<=n;j++) {
                temp=x[j];
                h=EPS*fabs(temp);
                if (h == 0.0) h=EPS;
                x[j]=temp+h;
                h=x[j]-temp;
                (*vecfunc)(n,x,f);
                x[j]=temp;
                for (i=1;i<=n;i++) df[i][j]=(f[i]-fvec[i])/h;
        }
        free_vector(f,1,n);
}/* fdjac() */

/*-----------------------------------------------------------------------------
  UTILITIES (from Numerical Recipes in C)
  ---------------------------------------------------------------------------- */
fcomplex Complex(float re, float im)
/* convert float variables to complex numbers */
{
        fcomplex c;
        c.r=re;
        c.i=im;
        return c;
}

fcomplex Cadd(fcomplex a, fcomplex b)
/* add complex variables */
{
        fcomplex c;
        c.r=a.r+b.r;
        c.i=a.i+b.i;
        return c;
}

fcomplex Csub(fcomplex a, fcomplex b)
/* subtract complex variables */
{
        fcomplex c;
        c.r=a.r-b.r;
        c.i=a.i-b.i;
        return c;
}

fcomplex Cmul(fcomplex a, fcomplex b)
/* multiply two complex variables */
{
        fcomplex c;
        c.r=a.r*b.r-a.i*b.i;
        c.i=a.i*b.r+a.r*b.i;
        return c;
}

fcomplex Cdiv(fcomplex a, fcomplex b)
/* divide two complex variables */
{
        fcomplex c;
        float r,den;
        if (fabs(b.r) >= fabs(b.i)) {
                r=b.i/b.r;
                den=b.r+r*b.i;
                c.r=(a.r+r*a.i)/den;
                c.i=(a.i-r*a.r)/den;
        } else {
                r=b.r/b.i;
                den=b.i+r*b.r;
                c.r=(a.r*r+a.i)/den;
                c.i=(a.i*r-a.r)/den;
        }
        return c;
}

float Cabs(fcomplex z)
/* determine the absolute value of a complex variable */
{
        float x,y,ans,temp;
        x=fabs(z.r);
        y=fabs(z.i);
        if (x == 0.0)
                ans=y;
        else if (y == 0.0)
                ans=x;
        else if (x > y) {
                temp=y/x;
                ans=x*sqrt(1.0+temp*temp);
        } else {
                temp=x/y;
                ans=y*sqrt(1.0+temp*temp);
        }
        return ans;
}

fcomplex Csqrt(fcomplex z)
/* determine the square root of a complex variable */
{
        fcomplex c;
        float x,y,w,r;
        if ((z.r == 0.0) && (z.i == 0.0)) {
                c.r=0.0;
                c.i=0.0;
                return c;
        } else {
                x=fabs(z.r);
                y=fabs(z.i);
                if (x >= y) {
                        r=y/x;
                        w=sqrt(x)*sqrt(0.5*(1.0+sqrt(1.0+r*r)));
                } else {
                        r=x/y;
                        w=sqrt(y)*sqrt(0.5*(r+sqrt(1.0+r*r)));
                }
                if (z.r >= 0.0) {
                        c.r=w;
                        c.i=z.i/(2.0*w);
                } else {
                        c.i=(z.i >= 0) ? w : -w;
                        c.r=z.i/(2.0*c.i);
                }
                return c;
        }
}

void nrerror(char error_text[])
/* Standard error handler */
{
        fprintf(stderr,"Run-time error...\n");
        fprintf(stderr,"%s\n",error_text);
        fprintf(stderr,"...now exiting to system...\n");
        exit(1);
}

float *vector(long nl, long nh)
/* allocate a float vector with subscript range v[nl..nh] */
{
        float *v;

        v=(float *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(float)));
        if (!v) nrerror("allocation failure in vector()");
        return v-nl+NR_END;
}

int *ivector(long nl, long nh)
/* allocate an int vector with subscript range v[nl..nh] */
{
        int *v;

        v=(int *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(int)));
        if (!v) nrerror("allocation failure in ivector()");
        return v-nl+NR_END;
}

fcomplex *cvector(long nl, long nh)
/* allocate a complex vector with subscript range v[nl..nh] */
{
        fcomplex *v;

        v=(fcomplex *)malloc((size_t) ((nh-nl+1+NR_END)*sizeof(fcomplex)));
        if (!v) nrerror("allocation failure in cvector()");
        return v-nl+NR_END;
}

float **matrix(long nrl, long nrh, long ncl, long nch)
/* allocate a float matrix with subscript range m[nrl..nrh][ncl..nch] */
{
        long i, nrow=nrh-nrl+1,ncol=nch-ncl+1;
        float **m;

        /* allocate pointers to rows */
        m=(float **) malloc((size_t)((nrow+NR_END)*sizeof(float*)));
        if (!m) nrerror("allocation failure 1 in matrix()");
        m += NR_END;
        m -= nrl;

        /* allocate rows and set pointers to them */
        m[nrl]=(float *) malloc((size_t)((nrow*ncol+NR_END)*sizeof(float)));
        if (!m[nrl]) nrerror("allocation failure 2 in matrix()");
        m[nrl] += NR_END;
        m[nrl] -= ncl;

        for(i=nrl+1;i<=nrh;i++) m[i]=m[i-1]+ncol;

        /* return pointer to array of pointers to rows */
        return m;
}

void free_vector(float *v, long nl, long nh)
/* free a float vector allocated with vector() */
{
        free((FREE_ARG) (v+nl-NR_END));
}

void free_ivector(int *v, long nl, long nh)
/* free an int vector allocated with ivector() */
{
        free((FREE_ARG) (v+nl-NR_END));
}

void free_cvector(fcomplex *v, long nl, long nh)
/* free a complex vector allocated with cvector() */
{
        free((FREE_ARG) (v+nl-NR_END));
}

void free_matrix(float **m, long nrl, long nrh, long ncl, long nch)
/* free a float matrix allocated by matrix() */
{
        free((FREE_ARG) (m[nrl]+ncl-NR_END));
        free((FREE_ARG) (m+nrl-NR_END));
}

/*------------------------------------------------------------------
  undef variables
  ------------------------------------------------------------------*/
#undef HORIZONTAL
#undef VERTICAL
#undef UPWIND
#undef DOWNWIND
#undef CROSSWIND
#undef Km2
#undef FREQ
#undef E1
#undef E2
#undef NAngle
#undef psiMin
#undef psiMax
#undef psiI
#undef SPECULAR
#undef MAXITS
#undef TOLF
#undef TOLMIN
#undef TOLX
#undef STPMX
#undef FREERETURN
#undef ALF
```

```
#undef TOLX
#undef TINY
#undef EPS
#undef FREERETURN
```

## DISTRIBUTION LIST

Integrating an Airborne L-band Sea Clutter Model into Research Laboratory Space Time Adaptive Processing (RLSTAP)

Poh Lian Choong

| | Number of Copies |
|---|---|
| **DEFENCE ORGANISATION** | |
| **Task Sponsor** | |
| **S&T Program** | |
| Chief Defence Scientist<br>FAS Science Policy<br>AS Science Corporate Management<br>Director General Science Policy Development | 1 |
| Counsellor, Defence Science, London | Doc Data Sht |
| Counsellor, Defence Science, Washington | Doc Data Sht |
| Scientific Adviser to MRDC, Thailand | Doc Data Sht |
| Scientific Adviser Joint | 1 |
| Navy Scientific Adviser | Doc Data Sht |
| Scientific Adviser, Army | Doc Data Sht |
| Air Force Scientific Adviser | 1 |
| Director Trials | 1 |
| **Aeronautical and Maritime Research Laboratory** | |
| Director, Aeronautical and Maritime Research Laboratory | 1 |
| Chief, Weapons Systems Division | 1 |
| Research Leader, Air Weapons System, Dr. D. Kewley | 1 |
| **Electronics and Surveillance Research Laboratory** | |
| Director, Electronics and Surveillance Research Laboratory | Doc Data Sht |
| Chief, Surveillance Systems Division, Dr. B.D. Ward | 1 |
| Chief, Electronic Warfare Division | 1 |
| Research Leader, Tactical Surveillance and Reconnaissance | 1 |
| Head, Surveillance Sensor Processing, Dr. J. Whitrow | 1 |
| Head, Surveillance Systems Modelling and Assessment, Mr. D. Fogg | 1 |
| Author: Dr. P.L. Choong, SSD | 3 |
| Mr. Vaughan Thomas, SSD | 1 |
| Ms. Tuyet Nguyen, SSD | 1 |
| **DSTO Research Library and Archives** | |
| Library Fishermans Bend | Doc Data Sht |

| | |
|---|---|
| Library Maribyrnong | Doc Data Sht |
| Library Salisbury | 1 |
| Australian Archives | 1 |
| Library, MOD, Pyrmont | Doc Data Sht |
| US Defense Technical Information Center | 2 |
| UK Defence Research Information Centre | 2 |
| Canada Defence Scientific Information Service | 1 |
| NZ Defence Information Centre | 1 |
| National Library of Australia | 1 |
| **Capability Systems Staff** | |
| Director General Maritime Development | Doc Data Sht |
| Director General Aerospace Development | Doc Data Sht |
| **Knowledge Staff** | |
| Director General Command, Control, Communications and Computers (DGC4) | Doc Data Sht |
| Director General Intelligence, Surveillance, Reconnaissance, and Electronic Warfare (DGISREW)R1-3-A142 CANBERRA ACT 2600 | Doc Data Sht |
| Director General Defence Knowledge Improvement Team (DGDKNIT) R1-3-A141, CANBERRA ACT 2600 | Doc Data Sht |
| **Navy** | |
| SO (SCIENCE), COMAUSNAVSURFGRP, BLD 95, Garden Island, Locked Bag 12, PYRMONT NSW 2009 | Doc Data Sht |
| **Army** | |
| Stuart Schnaars, ABCA Standardisation Officer, Tobruk Barracks, Puckapunyal, 3662 | 4 |
| SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), MILPO Gallipoli Barracks, Enoggera QLD 4052 | Doc Data Sht |
| **Air Force** | |
| **Intelligence Program** | |
| DGSTA, Defence Intelligence Organisation | 1 |
| Manager, Information Centre, Defence Intelligence Organisation | 1 |
| **Corporate Support Program** | |
| Library Manager, DLS-Canberra | 1 |

## UNIVERSITIES AND COLLEGES

| | |
|---|---|
| Australian Defence Force Academy Library | 1 |
| Head of Aerospace and Mechanical Engineering, ADFA | 1 |

| | |
|---|---|
| Deakin University Library, Serials Section (M List) | 1 |
| Hargrave Library, Monash University | Doc Data Sht |
| Librarian, Flinders University | 1 |
| **OTHER ORGANISATIONS** | |
| NASA (Canberra) | 1 |
| Info Australia | 1 |
| State Library of South Australia | 1 |
| Parliamentary Library of South Australia | 1 |
| **ABSTRACTING AND INFORMATION ORGANISATIONS** | |
| Library, Chemical Abstracts Reference Service | Doc Data Sht |
| Engineering Societies Library, US | 1 |
| Materials Information, Cambridge Scientific Abstracts, US | Doc Data Sht |
| Documents Librarian, The Center for Research Libraries, US | 1 |
| **INFORMATION EXCHANGE AGREEMENT PARTNERS** | |
| Acquisitions Unit, Science Reference and Information Service, UK | 1 |
| Library – Exchange Desk, National Institute of Standards and Technology, US | 1 |
| National Aerospace Laboratory, Japan | 1 |
| National Aerospace Laboratory, Netherlands | 1 |
| **TTCP SEN TP3 MEMBER** | |
| Mr. David DiFilippo, Defence Research Establishment Ottawa (DREO), 3701 Carling Ave, Ottawa, Ontario K1A 0Z4, Canada | 1 |
| Dr. Mark E. Davis, Air Force Research Laboratory (AFRL), Rome Research Site/SN, 26 Electronics Parkway, Rome, NY 13441-4514, USA | 1 |
| **Total number of copies:** | 49 |

Page classification: UNCLASSIFIED

| | |
|---|---|
| **DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA** | 1. CAVEAT/PRIVACY MARKING |
| 2. TITLE<br>Integrating an Airborne L-band Sea Clutter Model into Research Laboratory Space Time Adaptive Processing (RLSTAP) | 3. SECURITY CLASSIFICATION<br>Document (U)<br>Title (U)<br>Abstract (U) |
| 4. AUTHORS<br>Poh Lian Choong | 5. CORPORATE AUTHOR<br>Electronics and Surveillance Research Laboratory<br>PO Box 1500<br>Salisbury, South Australia, Australia 5108 |

| | | | |
|---|---|---|---|
| 6a. DSTO NUMBER<br>DSTO–TR–1206 | 6b. AR NUMBER<br>AR-012-013 | 6c. TYPE OF REPORT<br>Technical Report | 7. DOCUMENT DATE<br>September 2001 |

| | | | | |
|---|---|---|---|---|
| 8. FILE NUMBER<br>B9505-21-172 | 9. TASK NUMBER<br>DST 99/045 | 10. SPONSOR | 11. No OF PAGES<br>48 | 12. No OF REFS<br>13 |

| | |
|---|---|
| 13. URL OF ELECTRONIC VERSION<br>http://www.dsto.defence.gov.au/corporate/reports/DSTO–TR–1206.pdf | 14. RELEASE AUTHORITY<br>Chief, Surveillance Systems Division |

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

*Approved For Public Release*

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, SALISBURY, SOUTH AUSTRALIA 5108

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS

No Limitations

18. DEFTEST DESCRIPTORS

| | |
|---|---|
| Sea clutter | RLSTAP |
| L-band | Radar Modelling |

19. ABSTRACT

Research Laboratory Space Time Adaptive Processing (RLSTAP) is a compilation of a multiple-channel radar environment, signal and hardware programs integrated within the KHOROS environment. **RLSTAP** was initially developed for the US DARPA Mountain Top Program, and has since been used by several TTCP subgroups for collaborative efforts. Member nations of the SEN TP3 group have been using **RLSTAP** as the baseline radar system simulator for collaborative radar performance assessment. Under SEN TP3 Task 4 *Collaborative Radar Simulation Validation*, we have integrated an L-band sea clutter model with existing RLSTAP programs without extensive redesigning or recoding of the original sea clutter "C" program. This report serves to illustrate how the KHOROS software development tools can be used to modify a stand-alone L-band sea clutter model written in "C" for KHOROS compliance. The rest of the report contains an overview of **RLSTAP** clutter models and detail formulation of the second-order L-band sea clutter model.

Page classification: UNCLASSIFIED